Introduction

This technical note describes the FDM-TDM transmultiplexer, a class of digital signal processing algorithms used to construct a bank of digital filters. Such a filter bank can be used to separate the spectrally disjoint portions of an input signal. The need for such a filter bank arises frequently in practical applications and the growing capability of digital signal processing (DSP) devices makes the digital transmux approach ever more attractive compared to`alternate system designs.

This technical note begins by motivating the use of a digital filter bank for signal selection and processing applications. The section "Derivation of the equations for a Basic FDM-TDM Transmux" describes in analytical detail two classic, but distinct, ways of deriving the equations for an FDM-TDM transmultiplexer. Use of the resulting design equations is illustrated in the section "Example: Using an FDM-TDM Transmux to Demodulate R.35 Telegraphy Signals ". There it is explained how the algorithms are used to form the basis for a very efficient demodulator for frequency-shift-keyed (FSK) voice frequency telegraphy (VFT) signals.

The transmultiplexer rarely appears alone in signal sorting and processing applications and therefore it must be designed in coordination with other parts of the system. An important example of this is digital tuning of the input signal, an operation that frequently precedes the transmultiplexing operation. The section "The Impact of Digital Tuning on the Overall design of an FDM-TDM Transmux" examines the design interactions between these two functions and describes several examples of how these tradeoffs have been made in actual equipment.

Appendices are included that provide additional analytical details and a discussion of the issues associated with designing filter pulse responses for transmultiplexers.

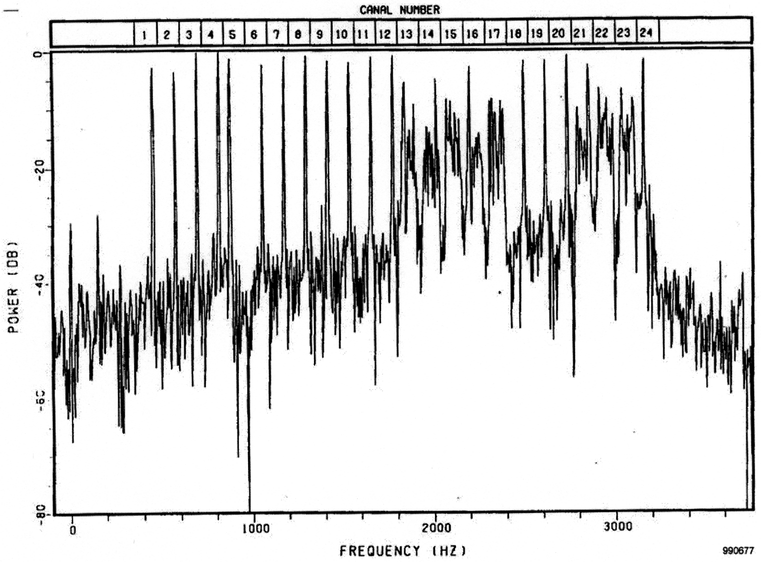
This document was originally written in 1989, a time when advancing semiconductor technology was first making it economically practical to use digital signal processing (DSP) concepts to build useful products. Because it was written so long ago, the reader will find the technology used in the examples to be overtaken by modern DSP products and devices. That noted, the mathematics captured here and the systems engineering trade-offs presented are still accurate and relevant for modern applications and implementations.

What is an FDM-TDM Transmultiplexer
Four kilohertz spacing is by far the most common
Canal Allocations for a Multichannel Voice Frequency Telegraph (VFT) Signal Conforming to CCITT Recommendation R.35 and a Typical Signal Spectrum

Frequency-Division Multiplexing

A common technique for sending many separate signals through the same physical medium is to use different portions of the available frequency spectrum for each one. Using spectral separation to permit the simultaneous transmission of signals from multiple users is generically called frequency division multiplexing (FDM). An example of this transmission technique is so-called FSK VFT. The spectrum of such a signal, along with its formal frequency allocations, is shown in [link]. In this case, designated the R.35 Recommendation by the ITU-T, each of the individual telegraphy signals is frequency-shift-keyed at a rate of 50, 60, or 75 bits/second and occupies one of 24 nonoverlapping spectral allocations within the 300 to 3400 Hz voice band. In the case of R.35, the *mark* and *space* frequencies are 60 Hz apart and the carrier, or center frequency, are 120 Hz apart.

The FSK VFT example will be returned to shortly. It should be noted first however that FDM techniques are widely used in telecommunications. An important example is multichannel FDM telephony in which many voice signals are bandlimited to about 3100 Hz each, single-sideband upconverted with carriers of different frequencies, and then summed. The resulting composite signal has spectrally disjoint channels at regular intervals of 3 or 4 kHz[footnote]. Even new fiber optic transmission systems are using FDM techniques, calling it instead wavelength division multiplexing (WDM).

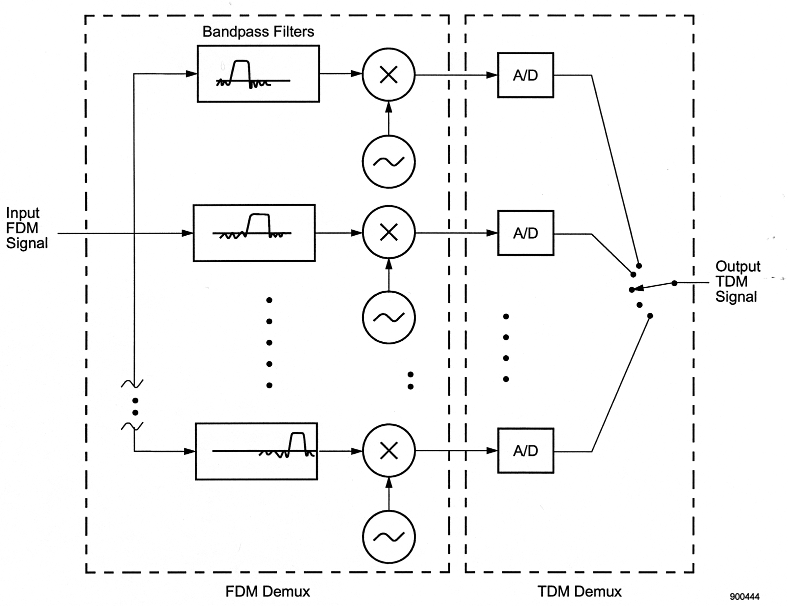


General Schematic of an FDM-TDM Transmultiplexer Composed of a Filter Bank and a TDM Multiplexer Fundamental Description of a Digital FDM-to-TDM Transmultiplexer

Use of a Filter Bank

Suppose now that we desired to separate the 24 individual telegraphy signals in an R.35 waveform so that each could be demodulated. A reasonable approach would be to build a bank of 24 filters to separate the individual FSK signals. A bank of 24 FSK demodulators would process the outputs of the filter bank. Note that in this case the filters need to be regularly spaced at intervals of 120 Hz and that each requires about the same bandwidth (about 90 Hz).

Suppose further that we desire to perform the demodulation digitally. This suggests the block diagram shown in [link]. The input FDM signal is applied to a bank of filters. Each filter has a bandpass characteristic centered on one of the 24 FSK canals. The filtered signals are then downconverted to a center frequency at or near DC and then digitized at a common rate high enough to satisfy the Nyquist sampling theorem for every FSK signal. We then choose to time division multiplex (TDM) the sampled FSK signals. This multiplexing allows all 24 signals to be placed on the same digital bus and perhaps to be processed by the same time-sharing digital demodulator.

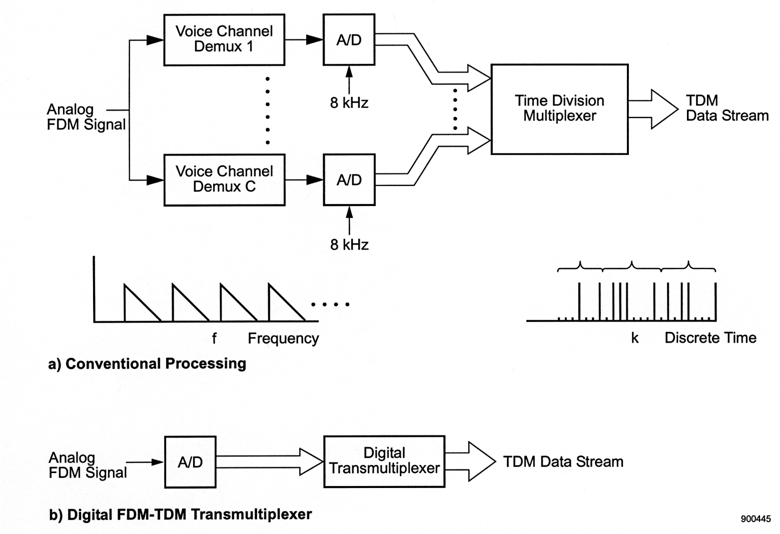


Looking again at [link] we see that the processing can be viewed as falling into five segments:

1. The filter bank
2. The downconversion
3. The sampling
4. The commutation of samples to produce a TDM bus carrying all signals
5. The demodulator, or more generally, the users of the individual sampled signals

While our objective was to separate the individual signals and to digitize them in preparation for possible processing, we observe at this point that steps 1 through 4 have the effect of converting the input FDM signal, in which each component signal is separated by frequency, into a TDM output signal, in which each component signal is available in its separate timeslot. This operation of converting from one form of multiplexing to another is termed transmultiplexing. The structure from FDM input to TDM output is therefore called an FDM-to-TDM transmultiplexer, or even more simply, an FDM-TDM transmux.

To this point no mention has been made of how the filter bank and downconversion process might be implemented. It could (and has) been done using analog filters and separate downconverters, each using its own local oscillator and mixer. This technical note describes algorithms that permit the same functions to be performed digitally. The conceptual distinction is shown in [link]. The top portion of [link] mimics the structure shown in [link]. The filtering and downconversion are performed discretely and then each output is digitized and commutated. The bottom portion of [link] shows the objective in the development of a digital FDM-to-TDM transmultiplexer. In this case, the input FDM signal is digitized. All band-pass filtering and downconversion is performed digitally. The downconverted outputs are then *read out* sequentially to produce the desired TDM output.

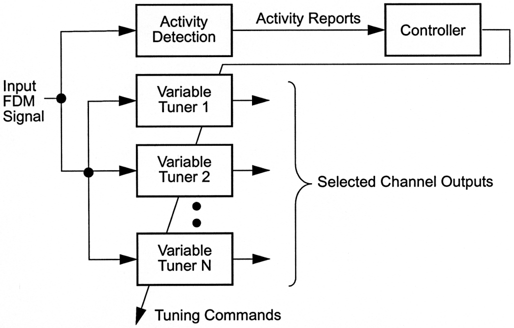


Two Methods of Processing Occasionally Active FDM Signals

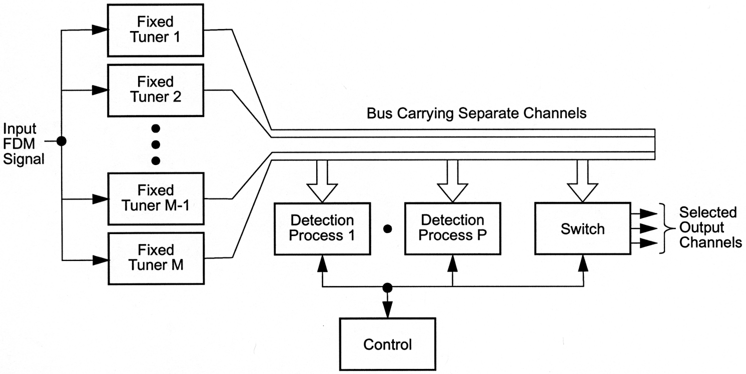
Processing Methods

We return to the example of demodulating the various FSK signals present in an R.35 VFT composite signal. Suppose that we use a transmultiplexer to separate the 24 FSK signals, or *canals*, as they are called, and place them on a TDM bus. Twenty-four demodulators or one time-shared demodulator convert the FSK signals into binary form. Thus the problem is neatly solved. In fact, the actual problem is slightly more complicated. In fact, only a small percentage of the 24 canals in a practical R.35 system are typically transmitting data at any given time. Most are in the *steady mark* or *steady space* condition. As a result, most of the 24 demodulators are unused at any given time. Is this concept of demultiplexing all of the canals the most efficient?

There are two basic and commonly used schemes for handling occasionally active FDM signals. Both are illustrated in [link]. The top scheme uses tunable filters and some common mechanism for detecting activity. Once activity is detected, a resource manager of some sort directs one of the tuners to the signal's frequency. The tuner output is then processed appropriately. In the case of FSK VFT, for example, the processor would be an FSK demodulator. The lower scheme is the one discussed earlier - all signals are demultiplexed and all processing, both activity detection and demodulation in the case of the VFT signals, is performed by using sampled waveform data taken from the TDM bus. In fact, systems have been built both ways, the choice depending on such factors as how the detector subsystem can be built, how many channels there are, how many signals might be active simultaneously, and the relative costs of implementation. The advent of the FDM-TDM transmultiplexer has shifted the balance toward the latter approach, particularly in applications where the activity factors are high or where several steps of processing are required, each of which needs independent and simultaneous access to the frequency channels.



a) Use of a Single Wideband Channel Activity Detector and A Few Variable Tuners



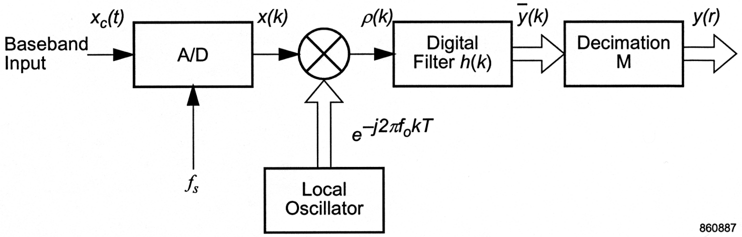
b) Use of a Bank of Fixed Tuners to Permit Nonblocking Per-channel Detection and Processing

900455

Derivation of the equations for a Basic FDM-TDM Transmux

Two intuitively reasonable approaches to developing the equations for the FDM-TDM transmultiplexer are presented in this section. The first emulates [link]. We first develop the equations for a digital counterpart of the analog tuners used in the filter bank and then observe that significant computational improvements can be obtained when the tuning frequencies are linked together in a simple way. The second subsection starts from a different point, that of using the discrete Fourier transform as a spectral channelizer. We ultimately find out that these two approaches yield essentially the same analytical results.

Using a digital Tuner to Extract One FDM Channel



Even though real-valued inputs are assumed here, all of the ensuing analysis applies to complex-valued signals as well. Spectral Description of Each Step in the Digital Tuning of a Single ChannelBecause of the K = 1 assumption, this equation is the simplest of all those seen to this point and will be referred to as the *basic equation*. Many applications require *M* to be chosen differently however (see Section 4 for example) and in these cases equation 12 should be used. Signal Flow to the Output of the Single-Channel Digital TunerAn important exception to this is the so-called prime-factor transform in which *N* is the product of small, prime factors (e.g., 2, 3, 5 , 7, 11, etc). The Number of Multiply-Adds Needed to Compute C Tuner Outputs for a Particular Set of System Parameters The Basic FDM-to-TDM Digital Transmultiplexer

The Transmux as a Bank of Single Channel Digital Tuners

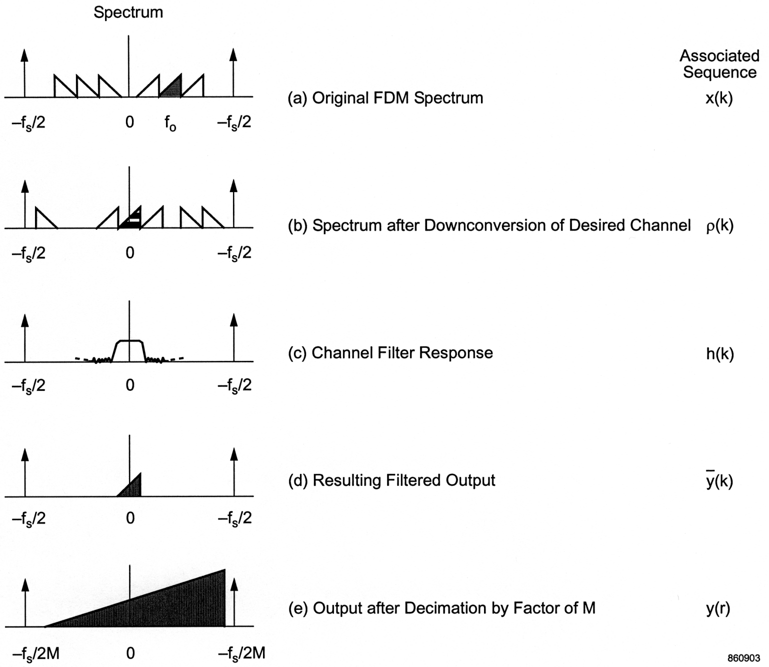
Fundamental equations for a Single-Channel Digital Tuner

The input FDM signal is assumed to be the continuous-time waveform xc(t). The analog-to-digital converter shown in [link] samples this waveform at the uniform rate of f_s samples per second, producing the discrete-time sequence x(k), where x(k)≡xc(t=kT), the integer *k* is the time index, and *T* is the sampling interval given by T=1fs. The spectrum of this sequence is shifted down in frequency by multiplying it by a complex exponential of the form e-j2πf0kT, where f_0 is the desired amount of the frequency downconversion. The product of x(k) and this exponential is then filtered in discrete time by using the pulse response h(k). The duration of the pulse response h(k) is assumed to be finite and in particular of length no greater than *L*, an integer. The filter output y¯(k) is then decimated by a factor of *M*, yielding the sequence y(r), where the integer *r* is the decimated time index.

These processing steps are shown in graphical form in [link]. Both sides of the two-sided spectrum of the sampled input signal are seen in [link](a). For the moment, the input signal is assumed to be real-valued and therefore the spectrum is symmetrical around 0 Hz [footnote]. A channel of interest in this spectrum has been shaded and its center frequency is noted to be *fo*. Multiplying the input signal by e-j2πf0kT has the effect of shifting the spectrum to the left (assuming 0≤f0≤fs2) and centering the desired channel at 0 Hz. The downconverted signal is now complex-valued, and therefore spectral symmetry around 0 Hz is neither required nor expected. The transfer function of the lowpass filter appears in [link](c). The filter pulse response h(k) is chosen to attain the desired spectral characteristics. In particular, the filter needs to pass the channel of interest without degradation and suppress all others sufficiently. How to design such a pulse response is discussed in Appendix A. In general, the quality of the filter grows with the value of the parameter *L*. The filter shown here is symmetrical around 0 Hz and its pulse response h(k) can therefore be real-valued. This is not required however.

After the application of the shifted signal ρ(k) to the filter, the spectrum shown in [link](d) results. The desired channel is isolated from all others. It is sampled, however, at a rate far faster than required by the Nyquist sampling theorem. The filter output is then decimated by the factor *M*, resulting in the spectrum shown in [link](e). The channel's bandwidth is the same as before but now its percentage bandwidth, that is, its bandwidth compared to its final sampling rate, is much higher. In a good digital tuner the percentage bandwidth after decimation usually ranges between 0.5 and 0.9, where unity is the theoretical limit imposed by the sampling theorem.

In principle, the parameters f_s (and hence T), f_0, L, and M can be chosen arbitrarily. In fact, significant simplications to the implementation of the tuner occur if they are carefully chosen. To do this we must first develop a general equation for the decimated tuner output y(r).


Spectrum
Associated Sequence
−f_s/2
0
f_o
−f_s/2
(a) Original FDM Spectrum
x(k)
−f_s/2
0
−f_s/2
(b) Spectrum after Downconversion of Desired Channel
ρ(k)
−f_s/2
0
−f_s/2
(c) Channel Filter Response
h(k)
−f_s/2
0
−f_s/2
(d) Resulting Filtered Output
ȳ(k)
−f_s/2M
0
−f_s/2M
(e) Output after Decimation by Factor of M
y(r)
860903

The undecimated filter output $y^{-}(k)$ can be written as the convolutional sum of $\rho(k)$ and the filter pulse response $h(k)$:
$y^{-}(k) = \Sigma l = 0 L - 1 h(l) \rho(k - l)$.

Substituting the expression for $\rho(k)$ yields
$y^{-}(k) = \Sigma l = 0 L - 1 h(l) x(k - l) e - j 2 \pi f 0 T(k - l)$.

Separating the two terms in the exponential produces the next expression:
$y^{-}(k) = e - j 2 \pi f 0 T k \cdot \Sigma l = 0 L - 1 h(l) x(k - l) e j 2 \pi f 0 T l$.

Decimation by the factor *M* is introduced by evaluating y¯(k) only at the values of *k* where k=rM. We denote the decimated output as y(r), given by

y(r) ≡ y¯(k = rM) = e-j2πj0TrM · Σl = 0L-1 h(l) x(rM-l) ej2πf0Tl

Choosing Various System Parameters to Simplify the General Equation for the Tuner Output

Equation 4 holds for arbitrary choice of *L*, *M*, f_0, and f_s. To obtain the equations for the basic FDM-TDM transmultiplexer, we must first simplify the general equation for the output of the digital tuner. We do this by making the three key assumptions:

1. We assume that the sampling rate f_s and the tuning frequency f_0 are integer multiples of the same *frequency step* Δf. In the case of FDM multichannel telephone systems for example, Δf is typically 4 kHz. We define the integer parameters *N* and *n* with the expressions fs≡N·Δf and f0≡n·Δf.
2. We next assume that the pulse response duration *L* is an integer multiple of the factor *N* defined above. We define the positive integer parameter *Q* where L≡Q·N. This is a nonrestrictive assumption since *Q* can be chosen large enough to make it true for any value of *L*. If QN exceeds the minimum required value of *L*, then h(k) can be made

artificially longer by padding it with zero values. The factor *Q* turns out to be an important design parameter. The parameters *Q* and *N* are determined separately and the resulting value of *L* follows from their choice.

3. We also assume that the decimation factor *M* is chosen to be closely related to the parameter *N*. Typical values are $M=N$ and $M=\frac{N}{2}$

We can now examine the effects of these assumptions. First, the relationship between f_s, f_0, and Δf allows y(r) to be written as

$$y_n(r) = e^{-j2\pi n r \frac{M}{N}} \cdot \sum_{l=0}^{L-1} h(l)\, x(rM-l)\, e^{j2\pi \frac{nl}{N}}.$$

We subscript the decimated output y(r) by the parameter *n* to indicate that it depends on the tuning frequency $f_0 = n \cdot \Delta f$.

The second assumption, the definition of the parameter *Q*, permits the single sum to be split into a nested double sum. To do this, define the new integer indices *q* and *p* by the expressions

$$l \equiv qN + p, \text{ where } 0 \leq q \leq Q-1 \text{ and } 0 \leq p \leq N-1.$$

Examination of [link] shows that the pulse response running index *l* has a unique value in the range from 0 to L-1 for each permissible value of *p* and *q*. This permits the single convolutional sum over the index *l* to be replaced (for reasons to be shown) with a double sum over the indices *p* and *q*. In particular,

$$y_n(r) = e^{-j2\pi n r \frac{M}{N}} \cdot \sum_{l=0}^{L-1} h(l)\, x(rM-l)\, e^{j2\pi \frac{nl}{N}} = e^{-j2\pi n r \frac{M}{N}} \cdot \sum_{p=0}^{N-1} \sum_{q=0}^{Q-1} h(qN+p)\, x(rM-qN-p)\, e^{j2\pi \frac{n(qN+p)}{N}} = e^{-j2\pi n r \frac{M}{N}} \cdot \sum_{p=0}^{N-1} \sum_{q=0}^{Q-1} h(qN+p)\, x(rM-qN-p)\, e^{j2\pi \frac{nqN}{N}} e^{j2\pi \frac{np}{N}} = e^{-j2\pi n r \frac{M}{N}} \cdot \sum_{p=0}^{N-1} e^{j2\pi \frac{np}{N}} \left[\sum_{q=0}^{Q-1} h(qN+p)\, x(rM-qN-p) \right].$$

The first portion of the exponential term in the sum vanishes since its argument is always an integer multiple of 2π. Moving the terms of the summation in the last step is possible since the remaining term of the exponential does not depend on the running index *q*. It is useful to give a short name to the terms in brackets in the last equation. Noting that it is a function of the decimated time index *r* and the running index *p*, we define the variable v(r,p) by the expression

$$v(r,p) \equiv \sum_{q=0}^{Q-1} h(qN+p) \cdot x(rM-qN-p).$$

Notice that v(r,p) is a function of the input data x(k), the filter pulse response h(l), and the system parameters *Q*, *M*, and *N*, but it is not a function of the selected conversion frequency *f0*, represented in the equation for yn(r) by the integer *n*.

Substituting v(r,p) into the equation for the decimated output yn(r) of the tuner tuned to

frequency f0=n·Δf yields
y n (r) = e - j 2 π n r M N · Σ p = 0 N - 1 e j 2 π n p N v (r , p)

Notice that the frequency dependency of the tuner shows up only in the exponential terms.

Before discussing this result in detail it remains to examine the effects of the third assumption. To do this, define the decimation factor *M* by the expression M≡NK, where K=1, 2, or 4. Look first at the exponential terms preceding the sum. It can now be written as
e - j 2 π n r M N = e - j 2 π n r K = [e - j 2 π K] n r = [- j 4 K] n r .

With *K* defined this way, the most general expression for yn(r) is
y n (r) = [- j 4 K] n r · Σ p = 0 N - 1 e j 2 π n p N v (r , p) , where
v (r , p) = Σ q = 0 Q - 1 h (q N + p) · x (r N K - q N - p) .

It can be verified that for K=1, 2, or 4, the factor multiplying the sum is at most a negation or a swapping from imaginary to real or vice versa. Thus no actual multiplication is needed. By far the cleanest case is the one in which the other system parameters (for example, *N*, *Q*, and h(k)) are selected so that the decimation factor M exactly equals N , or equivalently that K=1. In this case, the exponential preceding the sum collapses to unity, yielding what will be termed in this technical note as the basic FDM-TDM transmux equation[footnote]:

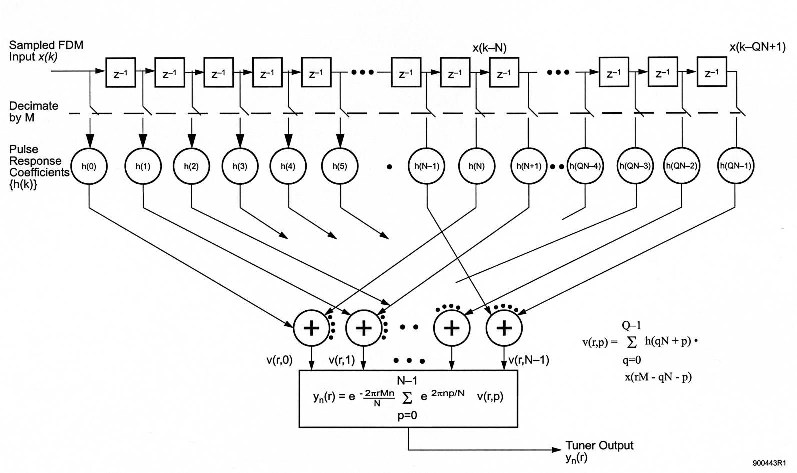
$y n (r) = \Sigma p = 0 N - 1 e j 2 \pi n p N v (r , p)$, where

$v (r , p) = \Sigma q = 0 Q - 1 h (q N + p) \cdot x ((r - q) N - p)$.

Interpretation of the Basic Tuner equation in Terms of the Discrete Fourier Transform

Examination of [link] shows that each sample of the tuner output, when tuned to frequency f0=nΔf, is the N-point inverse discrete Fourier transform (DFT) of the preprocessed data {v(r,p)}, evaluated at frequency index *n*. The signal flow described by the equation is shown in [link]. The sampled input data x(k) passes into a digital tapped delay line of length QN at the sampling rate f_s. Every M-th sample, the complete contents of the delay line, all QN samples, are used to compute {v(r,p)}. Thus the v(r,p) are computed at the decimated rate fsM. Each of the *N* elements of v(r,p) is computed by weighting *Q* of the delayed input samples by the appropriate coefficient from the pulse response vector h(k) and summing them together. Notice that at each decimated sampling interval all of the delayed data and all of the pulse response coefficients are used to compute the v(r,p). Notice also that since QN is usually much greater than *M*, each input sample is used in the production of the {v(r,p)} over several consecutive values of the decimated sampling index *r*.

The computation of the {v(r,p)} has several names in the literature. In some cases, it is referred to simply as the preprocessor or weighting processor. From the DFT-based filter bank interpretation of the transmultiplexer, in which the filter pulse function h(k) is viewed as a spectral window function, the operation is called windowing and folding. Some of the first researchers in the area [link] termed it *polyphase filtering*. Even though the reasons for this name are fairly obscure, it is commonly used.



Once the input data has been preprocessed, windowed and folded, or polyphase filtered, as you will, the resulting *N* values of v(r,p) are Fourier-transformed to produce yn(r). Notice that all of this computation must be repeated for each value of *r*.

It will be useful later to know how much computation is required to implement this *simplified* tuner. Assume for this calculation that the input data x(k) is complex-valued and that the pulse response h(k) is real-valued. If so, then 2QN multiply-add operations are needed for each computation of the {v(r,p)} and 4N multiply-adds (approximately) are needed for the computation of the single point of the DFT, all of this at the decimated sampling rate of fsM. A conventional tuner using a real-valued, L-point pulse response and complex input data requires 4fs multiply-adds for the mixer and 2fsLM multiply-adds for the filtering. Comparing the two shows that the filtering/weighting is exactly the same for the two, while the tuning vs. DFT comparison depends on the relative values of *M* and *N*. Using the example of the *basic transmux*, where N = M, we find that the two are equal. When M < N, the simplified equations actually require slightly more computation. Why then do we go to this trouble?

Generalization to the FFT-Based Digital Transmultiplexer

What if we desire to tune a second channel, say one that has a center frequency of fl = m·Δf ? Following through the derivation done before, we find that ym(r) is given by the same equations except that *n* is replaced with *m*. Examining the situation more closely we notice that the {v(r,p)} need not be recomputed to obtain the second tuner output. In fact, the only operation required to obtain the second tuner output is to recompute the inverse DFT, but this time evaluated for the index *m* instead of *n*. The conventional tuning approach must be completely repeated to obtain the output for another channel. It is usually the case that the computation of the {v(r,p)} is much larger than the computation required for the DFT. The fact that it need not be repeated quickly makes the preprocessor/DFT scheme significantly more efficient than the conventional digital tuner approach as the number of channels to be tuned grows. If we use the number of multiply-adds as an indication of computational complexity, and if we denote the number of channels to be tuned by the integer *C*, we can quantify this comparison by noting that

$$G_{\text{converntional}} = C[4f_s + 2f_sQNM] \text{ multiply-adds}$$

are needed for *C* conventional decimated digital tuners while

$$G_{DFT} = 2f_sQNM + 4Cf_sNM \text{ multiply-adds}$$

are needed for the preprocessor/DFT method.

The goal outlined in the section "What is an FDM-TDM Transmultiplexer" was to demultiplex all of the channels carried in the input FDM signal. If the input sampling rate is not chosen extravagantly, then the number of channels should be somewhat less than N2 if the input signal is real-valued, and somewhat less than *N* if the signal is complex-valued. To obtain the worst-case situation, we assume that it is complex-valued and that C=N. In this case, the total multiply-add computation is given by

$$G D F T (N \text{ channels}) = 2 f s Q N M + 4 f s N 2 M$$

.

Even though this value is less than that required by the direct tuning method, the quadratic dependence on the number of channels *N* makes this method expensive for situations where a large number of channels must be dealt with.

Solution to this problem comes in the form of the fast Fourier transform (FFT), a class of algorithms that can be used to efficiently compute all of the points of a DFT if *N* the size of the DFT, meets certain conditions. In particular, if *N* is a so-called *highly composite* number that is, it is the product of small positive integers, then various symmetries can be exploited to dramatically reduce the computation needed to compute the desired *C* tuner outputs.

In practice the size of the DFT, *N*, is typically chosen to equal *2R* or 4R2 , where *R* is some positive integer, resulting in what is known as the *radix-2* or *radix-4* FFT, respectively[footnote].

For discussion here we will assume the use of a radix-2 FFT (even though it is well known that the radix-4 algorithm is somewhat more computationally efficient). With this assumption we find that the number of multiply-adds needed to compute all *N* possible tuner outputs, is given by

$$G_{\text{radix-2 FFT}}(N_{\text{channels}}) = 2 f_s N M [Q + \log_2 N].$$

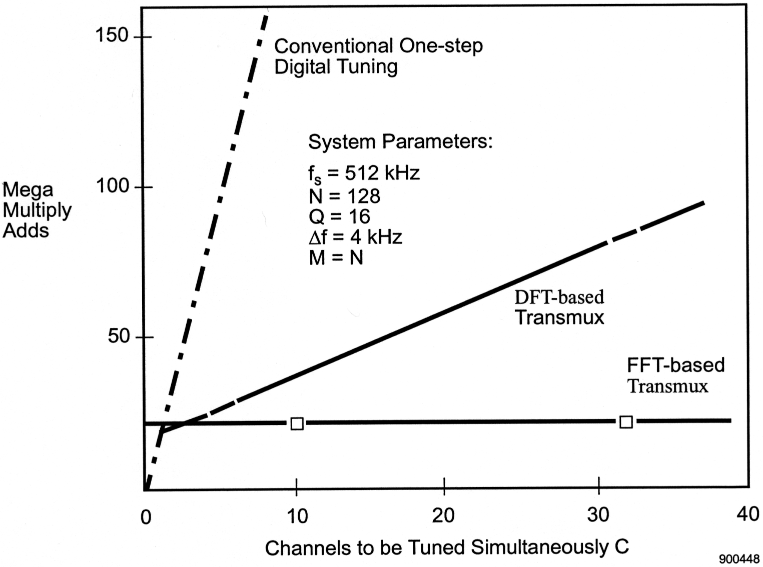
Comparison of this equation with [link] shows that the FFT-based method always requires less computation than direct DFT computation of all *N* tuners and requires less than the direct DFT computation of *C* tuners when *C* exceeds log2N. For example, suppose that: N=64 for a particular problem. If more than log264=6 tuners are required, then the FFT is more efficient. If *C* is more on the order of 50, as it probably would be, then FFT-based computation of the DFT is about eight times more efficient than direct computation of the DFT and even more efficient compared to conventional computation of the tuner outputs. A graphical example is shown in [link].

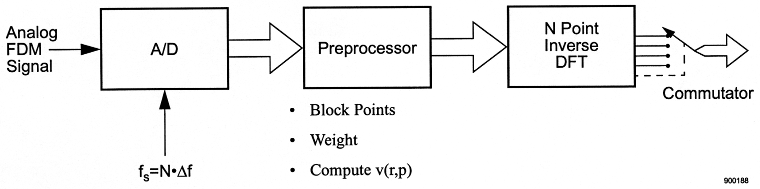
The generic FFT-based transmultiplexer consists of a preprocessor, which blocks, weights, and sums the input data to produce the *N* values of v(r,p), and an FFT, which efficiently computes the DFT for every value of *n*. This structure is shown in [link]. The input data is sampled (or provided by a preceding digital subsystem), preprocessed, and DFTed using the FFT algorithm. The FFT output bins are read out sequentially, thus producing the time division multiplexed (TDM) form promised originally.

The computational efficiency of the transmultiplexer can therefore be traced to two key items:

1. Separation of the tuning computation into two segments, one of which (the {v(r,p)}) need be computed only once
2. The use of the FFT algorithm to compute the inverse DFT

The first accrues from strategic choices of the sampling and tuning frequencies, while the second depends on *N* being chosen to be a highly composite integer.





Whether or not it is implemented with an FFT is irrelevant at this point. Also, we happen to use the inverse DFT to produce a result consistent with that found in the proceeding subsction, but the forward DFT could also be used. Processing Weighted, Delayed Signals with Discrete Fourier Transform Transfer Functions of the Paths from the Data Input x(k) to the DFT Outputs Xm(k) Comparison of the Unweighted Transfer Function Wm(ω) and a typical Desired Characteristic Effects of Changing Data Weighting and DFT Size Pruning a Decimation-in-Time (DIT) Fast Fourier Transform (FFT)

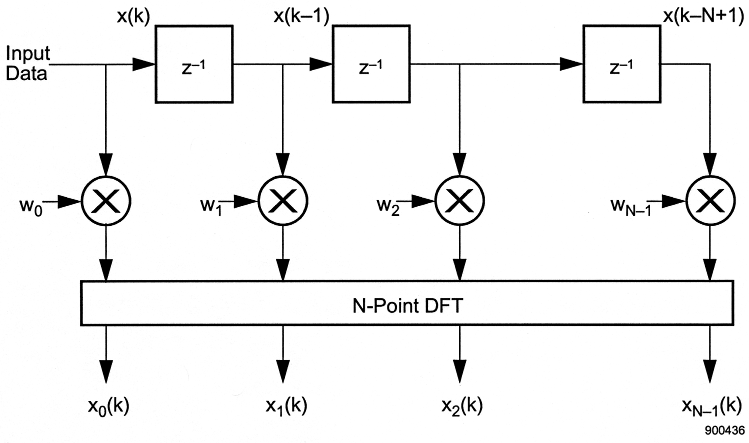
The Transmux as a DFT-based Filter Bank

We have just developed an FDM-TDM transmultiplexer by first writing the equations for a single, decimated digital tuner. The equations for a bank of tuners come from then assuming that (1) they all use the same filter pulse response and (2) their center frequencies are all integer multiples of some basic frequency step. In this section, we develop an alternate view, which happens to yield the same equations. It produces a different set of insights, however, making its presentation worthwhile.

Using the DFT as a Filter Bank

Instead of building a bank of tuners and then constraining their tuning frequencies to be regularly spaced, suppose we start with a structure known to provide equally-spaced spectral measurements and then manipulate it to obtain the desired performance.

Consider the structure shown in [link]. The sampled input signal x(k) enters a tapped delay line of length *N*. At every sampling instant, all *N* current and delayed samples are weighted by constant coefficients w(i) (where w(i) scales x(k-i), for *i* between 0 and N-1), and then applied to an inverse discrete Fourier transform[footnote]. The complete N-point DFT is computed for every value of *k* and produces *N* outputs. The output sample stream from the m-th bin of the DFT is denoted as Xm(k).



Since DFTs are often associated with spectrum analysis, it may seem counterintuitive to consider the output bins as time samples. It is strictly legal from an analytical point of view, however, since the DFT is merely an N-input, N-output, memoryless, linear transformation. Even so, the relationship of this scheme and digital spectrum analysis will be commented upon later. We continue by first examining the path from the input to a specific output bin, the m-th one, say. For every input sample $x(k)$ there is an output sample $X_m(k)$. By inspection we can write an equation relating the input and chosen output:

$$X_m(k) = \sum_{p=0}^{N-1} x(k-p)\, w(p)\, e^{j\frac{2\pi mp}{N}}$$

the m-th bin of an N-point DFT of the weighted, delayed data. We can look at this equation another way by defining $\bar{w}_m(p)$ by the expression

$$\bar{w}_m(p) \equiv w(p) \cdot e^{j\frac{2\pi np}{N}}$$

and observing that [link] can be written as

$$X_m(k) = \sum_{p=0}^{N-1} x(k-p) \cdot \bar{w}_m(p) .$$

From this equation it is clear $X_m(k)$ is the output of the FIR digital filter that has $x(k)$ as its input and $\bar{w}_m(p)$ as its pulse response. Since the pulse response does not depend on the time index *k*, the filtering is linear and shift-invariant. For such a filter we can compute its transfer function, using the expression

$$W_m(\omega) = \sum_{p=0}^{N-1} \bar{w}_m(p) \cdot e^{-j\omega pT} ,$$

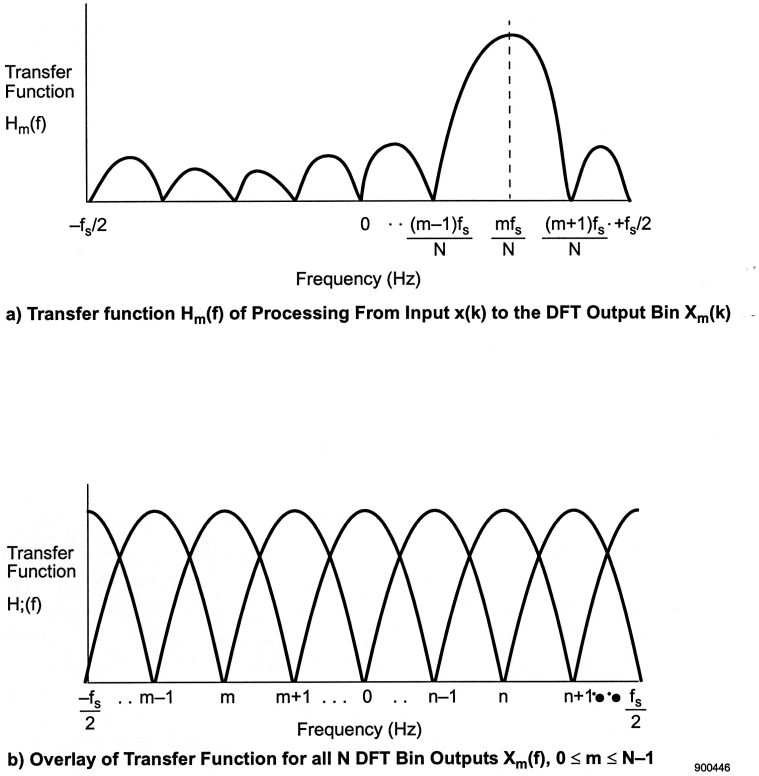
$$-\pi T \leq \omega \leq \pi T.$$

Suppose that we first choose the simple case with uniform weighting, that is, $w(p)=1$ for $0 \leq p \leq N-1$ and, therefore, $\bar{w}m(p)=ej2\pi mpN$. In this case, $Wm(w)$ is given by

$$W m(\omega) = e-j\pi mN \cdot e-j(N-1)\omega T2 \cdot \sin N\omega T2 \sin(\pi mN-\omega T2)$$

The magnitude of this transfer function is plotted in [link]. From this plot we can conclude that the pulse response $\bar{w}m(p)$ has what might be generally considered to be the frequency response of a bandpass filter. The filter is centered on bin *m* and its bandwidth is nominally fsN. While it might be characterized as a bandpass filter, we also note that the passband is quite rounded and the stopband rejection is relatively poor. The first sidelobes are only 13 dB lower than the peak of the passband response.

We've now shown that the path from the input to the m-th bin can be described as a finite impulse response (FIR) filtering operation and that the transfer function of that filtering operation has a fairly sloppy bandpass characteristic, at least when the data weighting is uniform. What happens for other values of *m* then? The answer is "the same thing." For each value of *m* between 0 and N-1, the pulse response $\bar{w}m(p)$ is computed, leading to the transfer function $Wm(\omega)$. An overlay of these bandpass responses is shown in the lower portion of [link]. From this we can conclude that the block diagram shown in [link] describes a single-input, N-output bank of filters. The filter center frequencies are spaced uniformly in increments of fsN Hz. All *N* outputs are sampled in time as frequently as the input. When the weighting function w(p) is uniform, then the bandpass filters have the form of [link], shown in [link].



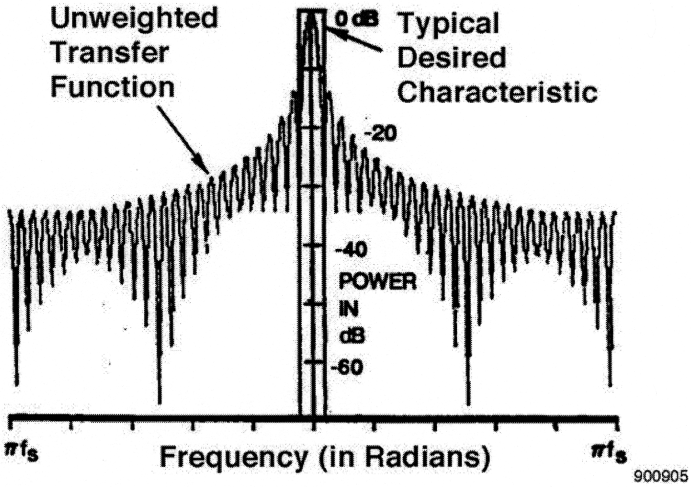
a) Transfer function $H_m(f)$ of Processing From Input x(k) to the DFT Output Bin $X_m(k)$

b) Overlay of Transfer Function for all N DFT Bin Outputs $X_m(f)$, $0 \le m \le N-1$

The Implications of Attaining the Desired Bandpass Characteristic

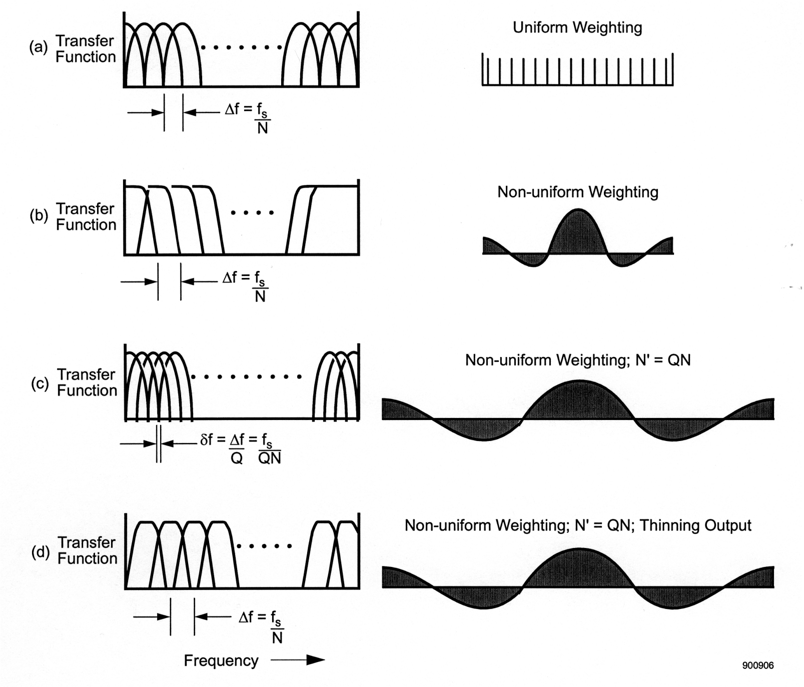
We've just shown that the DFT of delayed versions of the input sequence x(k) has the general properties of a bank of regularly-spaced bandpass filters. Two considerations leave us short of our goals. The first is that the shape of the transfer function for each bandpass filter is not good enough for most applications and must be improved. The second is that of reducing the amount of computation required. We address the first one in this section, and temporarily defer the computation issue.

[link] shows two transfer functions. The one pointed to from the left is exactly the same as that shown in [link] and defined in [link]. The one pointed to from the right is representative of the type needed for demultiplexing FDM multichannel telephone signals. It offers essentially flat response for most of the passband, has very sharp transition bands, and suppresses all energy outside of the transition bands by 55 dB or more. While other applications may require different transfer functions, as a rule they will be much more stringent than unweighted transfer function shown in [link].



How then do we attain different transfer function characteristics? In fact, we use some of the remaining degrees of freedom, the weighting function w(p). By allowing w(p) to be non-uniform we can now alter the shape of the transfer function of each bandpass filter. By using well-known FIR filter design techniques (see An Introduction to the FDM-TDM Digital Transmultiplexer: Appendix A) it is possible to attain virtually any shape. It is not, however, possible to always attain the desired shape and the desired bandwidth while keeping the duration of the pulse response constant. In fact, as discussed in An Introduction to the FDM-TDM Digital Transmultiplexer: Appendix A, for a constant bandwidth, the pulse response duration must grow as the transition bandwidth is forced to be smaller and as the stopband suppression in increased. The chain of events described in [link] then unfolds.

Shown across the top of [link] is a stylized version of that seen in the bottom portion of [link]. The uniform weight shown on the top right leads to the bandpass filter shapes shown on the left. Note that the filters are separated in frequency by fsN Hz.


(a) Transfer Function
Uniform Weighting
$\Delta f = \frac{f_s}{N}$
(b) Transfer Function
Non-uniform Weighting
$\Delta f = \frac{f_s}{N}$
(c) Transfer Function
Non-uniform Weighting; N' = QN
$\delta f = \frac{\Delta f}{Q} = \frac{f_s}{QN}$
(d) Transfer Function
Non-uniform Weighting; N' = QN; Thinning Output
$\Delta f = \frac{f_s}{N}$
Frequency
900906

Now suppose we employ non-uniform weighting to improve the shape of the bandpass filters. As discussed in Appendix A, such non-uniform weighting can be used to attain the desired transfer function shape, but virtually always at the expense of the bandpass filter's bandwidth. In fact, to obtain the desired characteristic shown in [link], with its flat passband, sharp skirts, and high-attenuation stopband, the minimum passband bandwidth is more than a factor of ten larger than the unweighted response. Thus the use of a non-uniform weighting, as shown on the right of [link](b), results in the situation shown on the left side. There are still *N* bandpass filters, and their center frequencies are still separated by integer multiples fsN Hz, but each filter has been widened considerably, leading to a high degree of overlap.

The first problem to deal with is not the overlap, but rather the fact that the individual bandpass filters are far wider than the original goal of about fsN Hz. This is dealt with by returning to [link] and simply letting the delay line length, the number of weighting coefficients, and the DFT order grow until the filters are sufficiently narrowband to meet our objectives. Again using the example of the desired frequency response seen in [link], the dimensions must grow by more than a factor of ten.

While the resulting dimensions can take on rather arbitrary values (above some minimum value) we'll assume here that the new size *N′* is an integer multiple of *N*. In particular, we assume that the delay line, and the weighting and DFT with it, are extended to the length *N′* where:

$$N' = Q \cdot N,$$

where *Q* is a positive integer. We further assume that *Q* is chosen to be large enough that a weighting function of length N' can be designed to produce not only the desired shape but also a bandwidth of about fsN Hz. The resulting situation is shown in [link](c). The weighting function is now longer than before (by a factor of *Q*). On the left we see that there are now N' filters in the filter bank. Each one of them now has the desired nominal bandwidth of fsN Hz, but their center frequencies are now separated by δf=fsN'=fsNQ Hz instead of fsN Hz. The overlap seen just above still exists but now there is a factor of *Q* more filters, a factor of *Q* narrower, and a factor of *Q* more closely spaced. Thus the positive effect of expanding the delay line dimension to QN is that the resulting filter bank includes the desired bandpass filters, both in bandpass characteristics and center frequencies. The negative aspects include the fact that the amount of weighting and DFT computation have gone up by a factor of *Q* and that there are now (Q-1)·N superfluous bandpass filters.

Suppose now that we choose to compute only every Q-th point of the DFT. The delay line is still QN samples long, there are still QN coefficients in the weighting function, and the DFT still has order QN, but we'll choose to only compute those output bins Xm(k) where *m* is an integer multiple of *Q*. This results in the situation shown in [link](d). The same QN-point weighting function is used as immediately above. This case, with *N* filters of nominal bandwidth fsN Hz and spaced fsN Hz apart, was our objective. To achieve it, however, required expanding the dimensions of the preceding operations quite considerably.

We now develop some equations that describe the steps just traversed. Starting with [link] we replace *N* with $N' = QN$, obtaining an expression for the time sequence seen at the m-th DFT bin.

$$X_m(k) = \sum_{p=0}^{QN-1} x(k-p)\,\omega(p)\,e^{j2\pi \frac{mp}{QN}}.$$

Suppose, as discussed above, that we eliminate the filter overlapping by evaluating only every Q-th DFT bin. Thus we compute $X_m(k)$ only for those values of *m* that are integer multiples of *Q*. Specifically, if *n* is assumed to be an integer, then we only compute $X_m(k)$ for values of *m* given by $m = Qn$. This is leads to

$$X_m(k) = X_{Qn}(k) \equiv X_n(k) = \sum_{p=0}^{QN-1} x(k-p)\,w(p)\,e^{j2\pi \frac{np}{N}}.$$

Since we have achieved the goal of constructing spectrally concentrated bandpass filters (albeit at the cost of expanding the size of all steps preceding the final DFT computation), we can now consider decimating the filter outputs. Since the filter bandwidths are nominally $\frac{f_s}{N}$ Hz, decimation by up to *N* is possible without violating the sampling theorem. Suppose we decimate by the factor *M*, where $0 < M \leq N$. This means evaluating the integer time index *k* only at integer multiples of *M*. If we allow the integer to be the decimated time index, the decimated version of the n-th DFT bin output is

$$X_n(k = rM) = X_n(r) = \Sigma_{p=0}^{QN-1} x(rM - p) w(p) e^{j2\pi \frac{np}{N}}.$$

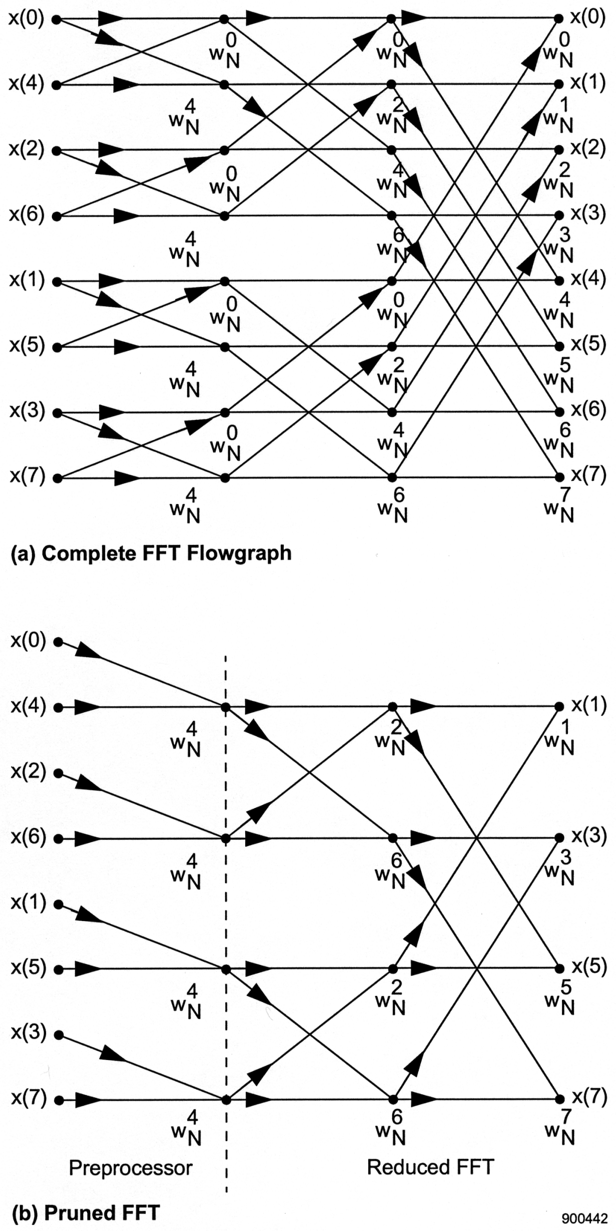
At this point we can start making comparisons. [link] closely resembles [link] and [link] closely resembles [link]. In fact, if we use the definition of v(r,p) developed earlier, then [link] becomes

$$X_n(r) = \Sigma_{p=0}^{N-1} v(r,p) \cdot e^{j2\pi \frac{np}{N}}$$

which differs from the equation for yn(r) developed in [link] only in the absence of a residual carrier term. If, for example, we want to compute yn(r), we can do it by selecting the right DFT output bin (*n* in this case) and multiplying it by the residual carrier term, if any. Thus for all practical purposes, the *bank of tuners* viewpoint and the *DFT-based filter bank* viewpoint yield the same structure and same results.

The Effect of Bin Decimation on an FFT

More insight into the relationship between the DFT-based filter bank and the basic FDM-to-TDM transmultiplexer shown in [link] can be gained by considering the common situation where an FFT is used to compute the DFT. In the preceding section, it was shown that the DFT filter implicitly uses a QN-point DFT but in fact only *N* output bins are computed. Consider now the FFT flow graph shown in [link](a). The input is QN (8, in this case) weighted input samples $\bar{x}(p)$ and the output is QN bins. Suppose now that all we want is the odd numbered output bins. Careful examination of the flow graph shows that more than just the output points can be deleted. Look at x(0), for example. It is computed using numbers from the previous stage which are only used to compute undesired outputs. Thus these intermediate terms need not be computed either. This process can continue until the point where the intermediate points are needed. To see how this works, examine [link](b). Removing all unneeded nodes reveals something very interesting. The FFT processing naturally breaks into two sections. The second section, the QN-point FFT pruned of all unneeded nodes, is recognized to have the flow graph of an N-point FFT. In fact, if the bin decimation is not offset from bin 0, then the *twiddle factors* are exactly those of an N-point FFT as well. The section preceding the N-point FFT can be written as *N* Q-point sums of weighted, delayed input data samples. These sums can be recognized as the {v(r,p)}. Thus by pruning out the unneeded nodes in a QN-point FFT taken over the weighted input data, the computation of the filter bank gracefully separates into the cascade of a preprocessor that computes the {v(r,p)} and an N-point FFT. The resulting block diagram is exactly the same as that shown in [link].



(a) Complete FFT Flowgraph

(b) Pruned FFT

Relationship of the Filter Bank Approach to

Digital Spectrum Analysis

A matter of confusion to many engineers is that the filter bank scheme seems to produce time samples from FFT bins. This confusion has its root in the fact that the DFT, and hence the FFT, are usually discussed in the context of digital spectrum analysis and are typically spoken of as methods of converting from the time domain to the frequency domain. How then can a DFT-based filter bank produce time samples from spectral bins? In fact, the right perspective is the opposite one.

Consider again [link] from the viewpoint of digital spectrum analysis. A simple FFT-based spectrum analyzer accepts *N* samples of the input sequence, weights or *windows* the data, transforms it by using an N-point FFT, and then estimates the power spectral density by computing the magnitude square of the bin outputs. Comparing these steps to [link], we see that they are identical except for two things: (1) the magnitude squaring operation at the bin outputs and (2) the fact that in spectrum analysis the window and transform operation is rarely done for every input sample. (Typically it is done every N-th sample [called *1:1 overlapping*] or even less frequently.) These facts suggest that DFT/FFT-based digital spectrum analysis is derived from the filter bank concept rather than the other way around. The filter bank shown in [link] uses a transform computed over a record of weighted, delayed input data to split the input signal's energy into *N* spectral bands. The degree to which this separation is completed depends on the choice of windowing or weighting function and on the length of the transform. If the function is chosen properly, the windowing operation and the DFT/FFT computation can be computed less frequently, that is, decimation can be introduced. In this context, the simple FFT-based spectrum analyzer can be recognized to perform an instantaneous power measurement at the output of each of the filters in the bank. The quality of the analyzer depends on the window function chosen and the DFT/FFT order *N* (as they affect the passband shape), the rate at which the filter outputs are computed (given by the decimation factor *M*), and the number of instantaneous power measurements averaged to obtain the spectral estimate. Thus we can conclude that the digital spectrum analyzer approximates the true power spectrum by measuring the power seen in each of the bandpass filter outputs produced by the DFT-based filter bank.

An interesting sidelight is that the most common name for the transmultiplexer preprocessor stems from the filter bank's relationship with digital spectrum analysis. Look again at [link]. The input to the QN-point FFT is QN weighted and delayed input samples. From the bank-of-tuners viewpoint we know that the weighting function w(k) is just the tuner pulse response h(k) needed to bandlimit the tuned signal properly. In the context of spectrum analysis, however, this function is called a data window. They are in fact identical and the tuner viewpoint provides the analytical basis on which to design the needed window. We've already observed that after pruning the FFT, the QN-point transform separates into two sections. The first section folds together *Q* windowed samples at a time to generate the N-point input to the FFT. From this viewpoint, it is commonly referred to as the window-and-fold section of the FDM-to-TDM transmultiplexer.

Stylized FORTRAN Implementation of a Basic FDM-TDM Transmux

Table 1 shows a stylized example of a software implementation of an FDM-TDM transmultiplexer. Some details of the initialization steps have been blurred for the sake of simplicity and the parameters user are certainly not those appropriate to all applications, but the code should serve as an accurate guide to the amount of computation needed and its organization.

```
SUBROUTINE TMUX(INPUT_ARRAY, INPUT_POINTER,
C
C   SUBROUTINE TMUX - IMPLEMENTS A BASIC FD
C   A NEW VECTOR OF CHANNELIZED CHANNEL OUT
C   "OUTPUT_VECTOR" IS PRODUCED EACH TIME T
C   UNLESS THE DATA IN "INPUT_ARRAY" IS EXP
```

```
C
    PARAMETER N = 64        !NUMBER OF CHANN
    PARAMETER Q = 3         !WEIGHTING FUNCT
    PARAMETER M = N         !DECIMATION FACT
C    THIS CHOICE OF M YIELDS BASIC TRANSMUX
C
    INTEGER M, N, Q, INPUT_POINTER, J, K, IN
    COMPLEX INPUT_ARRAY(1), OUTPUT_VECTORS(N
    REAL WEIGHTING(N*Q)
C
    DATA WEIGHTING/ * QN values of the weigh
C
C   ****************** MOVE TIME POINTER ***
C
    INPUT_POINTER = INPUT_POINTER + M
C
C   ****************** COMPUTE v(r,p) *****
C
    DO 10 J=1,N
10  VRP(J) = CMPLX(0.,0.) !ZERO THE VECTOR V
C
    DO 30 J=1,N
     DO 20 K=1 Q
      INDEX = (K-1)*N+J-1   !COMPUTE OFFSET
      VRP(J) = VRP(J) + INPUT_ARRAY(INPUT_PO
 1      WEIGHTING(INDEX+1)
20     CONTINUE
30  CONTINUE
C
C   ****************** COMPUTE VECTOR OF OUT
C   ****************** USING INVERSE FFT ROU
```

```
C
    CALL IFFT(VRP,OUTPUT_VECTOR,N)
C
C   ****************** REMOVE RESIDUAL CARRI
C
C   ------------ IF M NOT EQUAL TO N, REMOVE
C
C   ********************* NORMAL RETURN ****
C
    RETURN
C
    END
```

Stylized FORTRAN Example of an FDM-TDM Trans

Example: Using an FDM-TDM Transmux to Demodulate R.35 Telgraphy Signals

Suppose that our design objective is to build a digital processor capable of demodulating all of the FSK canals found in the R.35 signal shown in Figure 1 from "An Introduction to the FDM-TDM Digital Transmultiplexer: Introduction". Suppose further that we choose to build the demodulator for each FSK signal along the lines of the one shown[footnote] in [link](a). Both are used in practice. This type of FSK demodulator uses two fllters: one centered at the *mark* frequency fmk and another the *space* frequency fsp. The powers or amplitudes of the two filter outputs are compared to determine whether the signal instantaneously falls mostly in the vicinity of the mark or is closer to the space. The bit synchronizer logic monitors the transitions between mark and space (and vice versa), using the information to determine the right instants to sample the thresholded difference waveform and produce binary decisions.
Note that this demodulator design is slightly different than the one discussed in the section "What is an FDM-TDM Transmultiplexer".

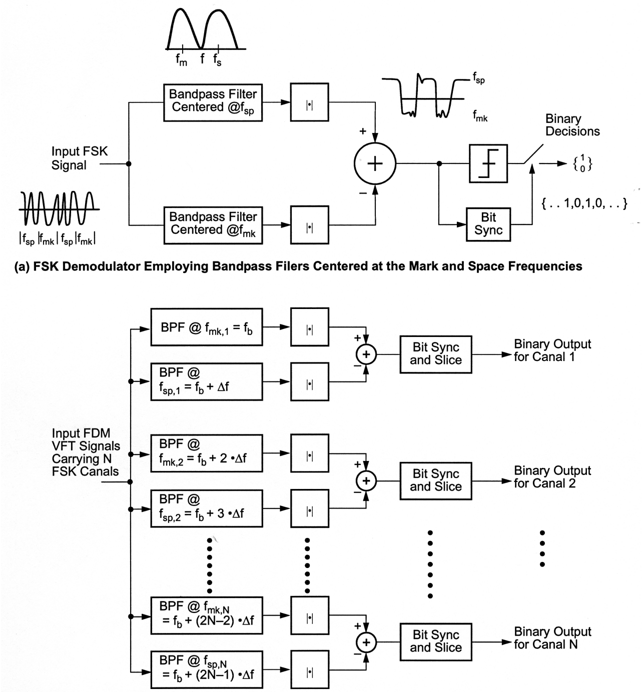
[link] shows the block diagram of the demodulating process when extended to handle all 24 FSK canals in an R.35 signal. Initially it appears to only be 24 parallel demodulators. On closer inspection however, it may be recognized that the center

frequencies of all the filters differ by integer multiples of a single frequency increment Δf. This suggests the use of a digital filter bank to compute all ofthe required bandpass filters. We now proceed to see how the system design for this filter bank is done.

The system design of the transmultiplexer/filter bank is specified by a small set of parameters. We determine these parameters as follows:

- Δf: Inspection of the frequency allocations for the R.35 signal shows that all possible mark and space frequencies are separated by integer multiples of 60 Hz. Thus it is natural to set $\Delta f = 60$ Hz.
- f_s and N: With Δf determined, the choice of N, the DFT dimension, and the choice of the input sampling rate f_s, are locked together. We bound N from below, by noting that at least 48 filters are needed, two for each FSK canal. In principle, the value of N can be chosen to be any number higher than 48. If the use of the FFT is contemplated then N is usually chosen to the first power of 2 or 4 higher than the minimum value[footnote]. Assuming the use of either a radix-2 or radix-4 FFT, plus the use of complex-valued input data, the prudent value of N is 64. This immediately leads to a complex-valued input sampling rate of $N \cdot \Delta f = 3840$ Hz. If the input were real-valued

instead, then the chosen sampling rate would be twice that, or 7680 Hz. A Basic Two-Filter FSK Demodulator and its Filter Bank Extension



f_m f f_s
Bandpass Filter Centered @f_{sp}
Bandpass Filter Centered @f_{mk}
|•|
f_{sp}
f_{mk}
+
−
Input FSK Signal
$|f_{sp}|f_{mk}|f_{sp}|f_{mk}|$
Binary Decisions
{1 0}
{. . 1,0,1,0, . . }
Bit Sync
Input FDM VFT Signals Carrying N FSK Canals
BPF @ $f_{mk,1} = f_b$
BPF @ $f_{sp,1} = f_b + \Delta f$
BPF @ $f_{mk,2} = f_b + 2 \cdot \Delta f$
BPF @ $f_{sp,2} = f_b + 3 \cdot \Delta f$
BPF @ $f_{mk,N} = f_b + (2N–2) \cdot \Delta f$
BPF @ $f_{sp,N} = f_b + (2N–1) \cdot \Delta f$
Bit Sync and Slice
Binary Output for Canal 1
Binary Output for Canal 2
Binary Output for Canal N

(a) FSK Demodulator Employing Bandpass Filers Centered at the Mark and Space Frequencies

(b) Extension to an FDM FSK VFT Signal by Using a Transmultiplexer to Perform the Required Bandpass Filtering

900457

[link] introduces some additional considerations in the design of digital systems in which the transmultiplexer is only a part.

- *L* and *Q*: Wth *N* determined, we find that *L* and *Q* are locked together and that they are a function of the exact filter design used to select the pulse response (or, equivalently, the window function) used to determine the shape of the bandpass fllters. The issues to be considered in the design of the pulse response are discussed in "An Introduction to the FDM-

TDM Digital Transmultiplexer: Appendix A". Without reiterating them here, we observe that following those rules yields a minimum pulse response duration *L* of about 174. For this application we extend the filter pulse duration to 192, allowing *Q* to equal exactly 3.

- *M*: With the input sampling rate set, the decimation factor *M* determines the output sampling rate at each of the filter outputs. Thus fout, the output sampling rate, equals 3840M Hz. The required output rate depends on the types of signals present and the types of processing to be done to them. In the case of demodulating asynchronous FSK signals, experience has shown that the output sampling rate needs to exceed the highest FSK baud rate expected by a factor of four or more. The highest baud rate allowed by the CCITT for an R.35 canal is 75 Hz[footnote]. Thus the output sampling rate fout must exceed 300 Hz. By choosing M = 12, we obtain an output sampling rate of fout = 320 Hz. The other rates are 50 and 60 Hz.

A demodulator using these parameters is shown in [link]. When developed in 1985 it was termed the *filter bank* card and was used to simultaneously demodulate 24 voice grade channels, each containing 24 R.35 FSK canals. Thus 24·24·2 = 1152 fllters were needed to isolate the mark and space energy of all FSK canals. When demodulating R.35

signals, each filter produces outputs at a rate of 320 per second. The input sampled waveforms are tuned and filtered by a preceding tuner to spectrally align the filter bank's bins with the mark and space frequencies of the R.35 signal.

Arrows in [link] point to the key computational elements on the filter bank card. One multiplier-accumulator device handled the window-and-fold, or preprocessing, function for all 24 voice channel inputs. The second computed the needed FFTs. When processing R.35 signals in all 24 channels, 320·24 = 76800 FFTs of dimension N = 64 were performed per second. The third arrow points to a floating-point processor used to measure the instantaneous power at the bandpass filter outputs and threshold their differences to produce binary decisions.

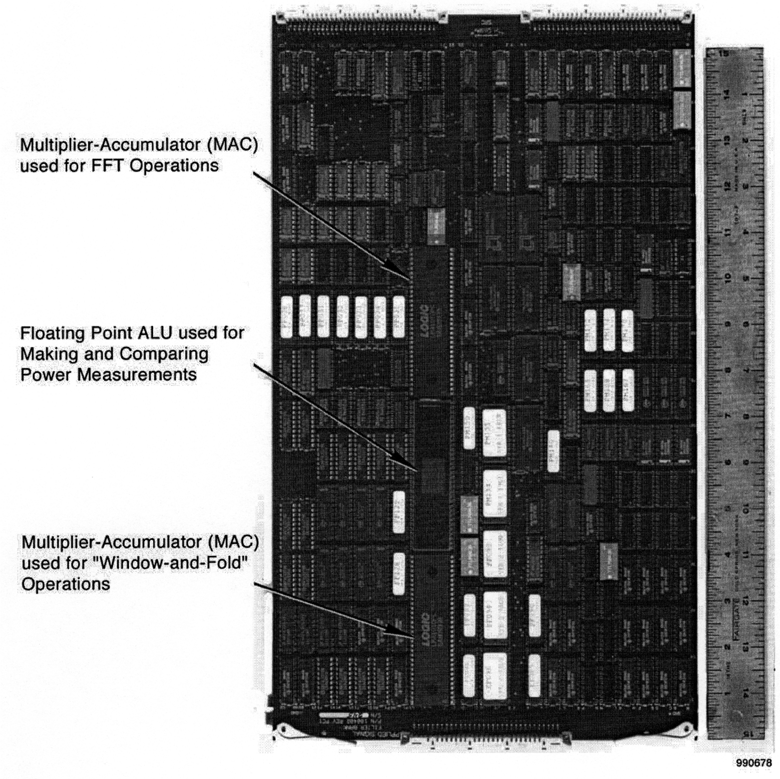
In passing, we note that this design exploits the fact that the FSK demodulator only requires knowledge of the magnitude of each filter output. Recalling the general transmux equation given in Equation 9 from "Derivation of the equations for a Basic FDM-TDM Transmux" and substituting the appropriate values of N,Q, and *M* produces the equation:

$$y_n(r) = e^{-j\frac{2\pi 12 \cdot n r}{64}} \cdot \sum_{p=0}^{63} e^{j\frac{2\pi n p}{64}} v(r,p).$$

The exponential preceding the sum has unity magnitude and therefore does not affect $|y_n(r)|$.. It is therefore not necessary to compute the product of the exponential and the sum. The magnitude measurement is given by

$|yn(r)| = |\Sigma p = 0\ 63\ ej2\pi np\ 64\ v(r,p)|$

. Photograph of the Filter Bank Card Designed in 1985 to Demodulate FSK VFT Signals - M=12, Q=3, N=64, Δf=60 Hz. A modern implementation would require a fraction of the resources of a single field-programmable gate array (FPGA)


Multiplier-Accumulator (MAC) used for FFT Operations
Floating Point ALU used for Making and Comparing Power Measurements
Multiplier-Accumulator (MAC) used for "Window-and-Fold" Operations
990678

This same filter bank card is also capable of demodulating FSK signals conforming to the R.37 and R.38A ITU-T recommendations. The R.37

signal, for example, uses 12 canals instead of 24, and each of them operates at twice the rate and with twice the mark-space frequency separation of the R.35 signal. Repeating the system design just performed yields the following:

- $\Delta f = 120$ Hz
- $N = 32$ and $fs = 3840$ Hz
- $L = 96$ and $Q = 3$
- $fout = 640$ Hz

Note that *Q* and *fs* remain the same as for R.35, and that Δf and fout double because of the increased mark-space separation and allowable baud rates, while *L* and *N* are halved. The impact of processing this additional signal can be assessed by using the formula for the number of multiply-adds required, specifically
$G \text{ transmux} = 2 Q N f s M + 2 N f s M \cdot \log 2 N .$

It can be verified that the number of multiply-adds needed to perform the window-and-fold function for the filter bank is exactly the same as is needed for the R.35 signal, since the ratio NM holds constant. Moreover, the amount of computation needed for the radix-2 FFT is 56 of that needed for R.35, the ratio between log 232 and log 264. Thus a filter bank with the computational horsepower to handle R.35 can also handle R.37. The R.38A standard represents another factor of two in frequency separations and allowable baud rates. Once again it can be verified that the window-and-fold computation is the same and the FFT computation is smaller yet. Thus a properly designed filter bank processor capable of handling the R.35 standard can also handle R.37 and R.38A as well. A subtle difference is that the input signal must be tuned slightly differently for the three different standards.

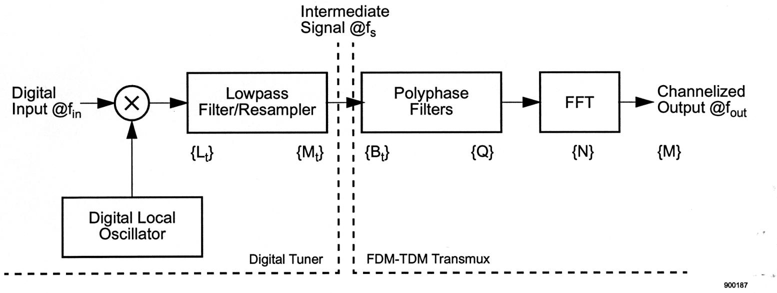
The Impact of Digital Tuning on the Overall design of an FDM-TDM Transmux

Problem Statement

So far we have presumed that the FDM input signal to the transmux has been magically provided and that it has been sampled at the proper rate. In fact, the signal available to the processor might not be in the desired form and signal processing may be required to convert it appropriately. As we shall see, the computation required for this can be significant in itself. As a result, these signal conditioning steps must be taken into account in the optimal design of the whole system. In this section, we focus on the use of digital tuners for this signal conditioning and examine the tradeoffs between the parameters of a tuner and the transmultiplexer that follows it.
We assume one-step decimation in this analysis. An important exception to this approach is described in Section 5.4.3. Key Variables in the Size Optimization of a Digital Tuner and TransmultiplexerSpecifically, at = 0.22 + 0.0366·SBR, where *SBR* is the minimum stopband rejection in decibels. A typical value for *SBR* is 60 dB, yielding an α_t of 2.42.

Total Computational Requirements

There are a few practical applications in which the input signal is complex-valued, sampled at the desired rate, and spectrally registered with the filters produced by the transmux-based filter bank. More typically, however, applications involve real-valued input signals, the signal is not aligned with the filters in the bank, or the signal of interest must be extracted from a wideband signal. It is common in these cases to use a digital tuner to select the portion of the spectral band in which the transmux will operate. This tuner will usually have a block diagram exactly like that seen in Figure 1 from "Derivation of the equations for a Basic FDM-TDM Transmux". The incoming sampled signal is quadrature downconverted, filtered using an FIR linear phase filter, and then decimated[footnote]. The decimated tuner output is applied to the preprocessor portion of the transmultiplexer. For the analysis here we assume that the input is real-valued (from an A/D converter, for example), that the tuner input sampling rate is given by fin, that the pulse response duration of the tuner's filter is given by L_t and that its decimation factor is M_t. The spectral band over which the tuner offers rated passband performance and adjacent signal rejection is denoted by B_t. The combined block diagram of the tuner and FDM-TDM transmultiplexer is shown in [link], along with the key variables needed to determine the joint optimal design.



We obtain an equation for the total number of multiply-adds required by adding the transmux expression found in Equation 18 from "Derivation of the equations for a Basic FDM-TDM Transmux" with the computation requirements of the preceding tuner. This produces the following:

$$G_{total} = G_{tuner} + G_{transmux}$$

$$= 2f_{in}\{1 + L_tM_t\} + 2f_sNM \cdot \{Q + \log_2 N\}.$$

By inspection we see that $f_{in}M_t = f_s$ and that $f_{out} = f_sM = f_{in}M_tM$.

We observe that the bandwidth of the signal exiting the tuner, denoted B_t, must be less than f_s, the transmux input rate, in order to satisfy the Nyquist sampling theorem. Their ratio is a key element in the computational tradeoff between the tuner and the transmux. With B_t fixed, an increase in f_s increases the computation needed for the transmux while decreasing that needed for the tuner. We make this explicit by developing a formula for the tuner's pulse response duration L_t. Again assuming one-step decimation and appealing to the design formulas discussed in [link], L_t is closely approximated by

$$L t = \alpha t f i n \Delta f t ,$$

where α_t is determined by the degree of stopband rejection desired[footnote] and Δft is the tuner's transition band. In this case, the transition band can be no greater than the difference between B_t and f_s. If we assume the use of this limiting value, L_t is given by

$$L t = \alpha t f i n f i n M t - B t$$

Substituting this expression and expressing all sampling rates in terms of the input rate fin produces an equation for the total number of multiply-adds required.

$$G \text{ total} = 2 f i n \cdot \{ 1 + \alpha t f i n f i n - M t B t \} + 2 f i n N M M t \{ Q + \log 2 N \} = 2 f i n \cdot \{ 1 + \alpha t 1 - (M t B t f i n) + N M M t \{ Q + l o g 2 N \} \}$$

Another useful form of this equation makes the functional dependence on f_s more explicit. We do this by using the expressions

$$N \equiv f s \Delta f , M t \equiv f i n f s , \text{ and } K \equiv N M$$

and the assumption that a radix-2 FFT is employed to compute the DFT. With these, the expression for the total number of multiply-adds can be written as

$$G \text{ total} = 2 f i n + 2 \alpha t f i n 1 - B t f s + 2 f s K \cdot \{ Q + l o g 2 f s \Delta f \}$$ Tradeoff between the Design

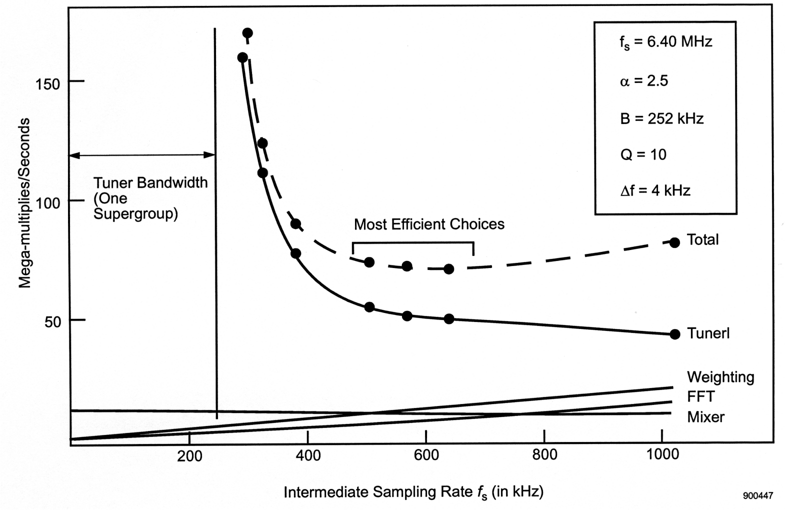
Parameter f_s and Total Computation in a Hypothetical Supergroup Transmultiplexer

Parameter Optimization

Given expressions such as those shown in [link] and [link] it is possible to accurately estimate the total amount of multiply-add computation needed for a tuner/transmux processor. It is also possible to perform tradeoffs between the various parameters in order to optimize the resulting design. While this can in principle be done with any of the design parameters, we demonstrate in this section the computational implications of varying the parameter f_s, the input sampling rate to the transmultiplexer. In practice, this usually turns out to be one of the designer's most important parameter choices.

[link] shows the computational requirements for a hypothetical transmultiplexer. In this case, the input sampling rate fin is assumed to be 6.4 MHz. The tuner must select an FDM telephone supergroup from the input signal and demultiplex all 60 voice channels in the supergroup. The tuner's bandwidth B_t must therefore be greater than or equal to 240 kHz and f_s must exceed that. For the telephone demultiplexing application, the channel spacing Δf is usually 4 kHz and the *over-sampling factorK* is typically chosen to be unity. [link] shows five curves, one for each segment of the computation

and one for the composite. The number of multiply-adds required by the input mixer is constant, since the input sampling rate fin is fixed. The computation required by the tuner's filter falls as *fs* rises from 240 kHz and tends toward the input Nyquist frequency of 3.2 MHz. The cause of this can be ascertained by examining [link]. As *fs* decreases toward *Bt*, the transition band decreases, *Lt* increases hyperbolically, and the amount of computation needed for the tuner's filter grows without bound.



The next two curves describe the effect of *fs* on the two components of the transmultiplexer. For a given value of *Q*, the computation required by the preprocessor is strictly proportional to *fs*. The FFT's computation rises slightly faster than proportionally since the number of FFT bins grows as *fs* does. The sum of these constituent curves represents the total amount of multiply-add computation needed. Note that it has a broad minimum. It rises precipitously as f_s decreases toward B_t and more slowly as f_s increases toward its other limit fin2.

The value of f_s which leads to the minimum amount of computation is a complicated and nonlinear function of virtually all of the design parameters. While an exact closed form equation for this minimum point is not attainable, it is possible to develop a useful approximation. We now proceed to do that.

We have made various assumptions about f_s along the way, the most important being that it is an integer multiple (and usually a power-of-two multiple) of the filter bank's channel separation Δf. For this analysis, however, we temporarily release that constraint and treat it as a continuous variable. To find its optimal value we can then evaluate the first derivative of G_{total} with respect to f_s and then find the value of f_s which makes the first derivative equal to zero. We first find that the derivative is given by

$$\frac{dG_{total}}{df_s} = \frac{-2\alpha_t f_{in,B_t}}{(f_s - B_t)^2} + 2K(Q+1) + 2K \cdot \log_2 \frac{f_s}{\Delta f}.$$

Setting the derivative to zero leads to an implicit, nonlinear expression. While it can be solved numerically, a practically valid assumption allows a closed form solution. We first define the variable γ, given

$$\gamma = \frac{K\{(Q+1) + \{\log_2[\frac{f_s}{\Delta f}]\}\}}{\alpha_t}.$$

With this definition we can write the equation determining the optimum point as

$$f i n B t (f s - B t) 2 = \gamma .$$

For convenience, we also define the factor ρ, a function of the tuner bandwidth reduction ratio, by $\rho = finBt$. Using this definition, [link] can be compactly, but deceptively, written as

$$(f s) \text{optimum} = B t (1 + \rho \gamma) .$$

This expression is deceptive since it proves to be implicit. The term γ depends on f_s, keeping [link] from being easily solved exactly. However, the equation proves to be useful anyway. Examination of the definition of γ shows that it depends on the logarithm of f_s and, in fact, is often quite insensitive to the actual choice of f_s. Once a general range of f_s has been determined, a nominal value of γ can in turn be found and plugged into [link] to find a value of f_s very close to the unconstrained optimum.

We can use the hypothetical supergroup tuner/ transmux to demonstrate this procedure. Suppose we guess the optimum value of f_s to be 480 kHz, twice the required tuner bandwidth B_t of 240 kHz. Plugging this into the expression for γ yields 10.4 and using that in [link] indicates that the optimum value for f_s should be about 625 kHz. [link] shows the curve to be quite flat in the vicinity of the optimum point, allowing the actual value of f_s to be chosen consistently with some of the constraints so far ignored in this analysis. In particular, we desire f_s to be a power of two or four times the channel spacing of 4 kHz in this case. Thus a reasonable choice for f_s in this case is 512 kHz.

We can observe some general trends affecting the optimal choice of f_s. It grows higher as the tuner input sampling rate fin does, reflecting the associated growth in tuner computation. It tends to decrease with growth in Q, K, and N, all of which imply more computation in the transmultiplexer. We note also that this formula depends strongly on the assumption of one-step decimation in the tuner. If a multistage tuner is used, the balance will be different. A rule of thumb can be developed by using [link]. Over a broad range of practical examples,the optimal ratio between f_s and B_t attains values between 1.3 and 2.3 for one-stage decimation. When this ratio (that is, $1+\rho\gamma$) exceeds 2.5 or so, the tuner computation overwhelms that of the transmux and alternative designs for the tuner should be examined. Multistage decimation is only one possible alternative. [link]

One implication of f_s being significantly larger than B_t is that many of the channels or filters in the transmux-based filter bank are not useful. To visualize this, consider [link]. [link](a) shows the power transfer function of the tuner filter before its output is decimated to the rate f_s. The passband of the filter is B_t Hz wide, the transition band on each

side of the passband is Δft Hz wide, and the stopband extends from Bt2 + Δft Hz to the Nyquist folding frequency fin2. [link](b) shows the power transfer fumction of the decimated filter. In this case, we assume that the transition band Δft is slightly less than fs-Bt. With this choice, some energy passed by the tuner through the transition bands folds back into the output, but none falls in the passband. [link](c) shows the channels of the transmux-based filter bank overlaying the tuner's power transfer function. The channels falling within the passband are clean, that is, the tuner's passband ripple and stopband rejection apply there, but the channels falling in the transition band are subject to several degradations (for example, gain slope and out-of-band signal aliasing) and are therefore not useful in most cases. Thus even though the transmultiplexer breaks the *fs* Hz band at the output of the tuner into *N* channels, only C of them, where C = BtΔf = N·Btfs, are typically used for downstream processing.
The Impact of the Tuner's Transition Bandwidth on the Number of Useful Filterbank OutputsThis demodulator was also capable of digitizing real-valued analog inputs at a rate of 16 kHz.
Photograph of a Supergroup Tuner and Transmultiplexer [link], circa 1985- M = 32, Q = 16, N = 128, Δf = 4 kHzThe zero-filling factor is 6 for input signals sampled at 16 kHz. The Use of a Resampling Tuner to Provide the Inputs to FSK VFT Demodulator "Filter Bank" Card Block Diagram of a

Quad Group Transmux ASIC

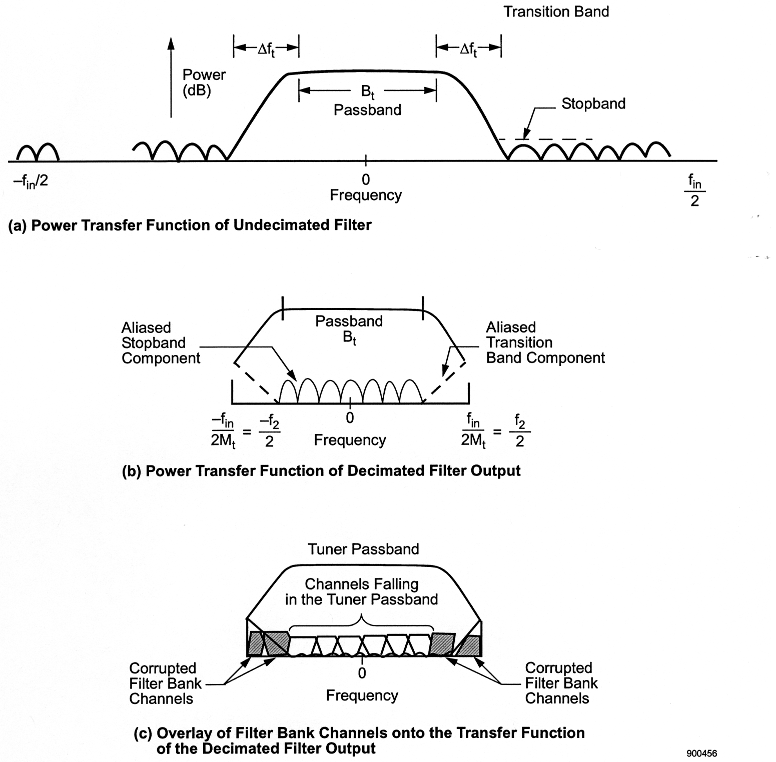
Hardware Examples of Tuner/Transmux Tradeoffs

This company has built a number of digital transmultiplexers for various applications and all of them employ some form of digital tuner. The next three sections present a few of these designs with the intent of demonstrating how the overall system design decisions were made.

A Single-Card Supergroup Tuner/Transmux

As a part of an IR&D program, the company developed an FDM supergroup transmultiplexer during 1985. Its basic requirements were to accept an FDM supergroup (that is, 60 voice grade channels spaced at regular intervals of 4 kHz over a band of 240 kHz) located at any of several possible spectral bands. These bands include 2-242 kHz, 12-252 kHz, 60-300 kHz, 312-552 kHz, and 564-804 kHz. Another key goal was excellent technical performance. To achieve this, the transmultiplexer portion was designed to use 16-bit arithmetic and key design parameters of fs = 4 kHz, K = 1, and Q = 16.

Since a supergroup only occupies 240 kHz, a convenient choice of f_s would be 256 kHz. This value exceeds 240 kHz and makes *N* equal 64, an integer power of two and four. This value proves not to be globally optimum, however, as we will see after examining the tuner's requirements.



(a) Power Transfer Function of Undecimated Filter

(b) Power Transfer Function of Decimated Filter Output

(c) Overlay of Filter Bank Channels onto the Transfer Function of the Decimated Filter Output

The highest input frequency of interest to the tuner is 804 kHz. The sampling rate must therefore exceed this value by two or more. The actual rate chosen was 2.048 kHz. This was based on several considerations:

- It satisfies the Nyquist sampling theorem and includes some allowance for the imperfections of analog antialias filtering.

- It is a power-of-two integer multiple of Δf = 4 kHz.
- It was the highest sampling rate attainable with financially acceptable 12-bit A/D converters of the era. Twelve-bit digitization was desired to maximize the unit's noise power ratio (NPR) and dynamic range.

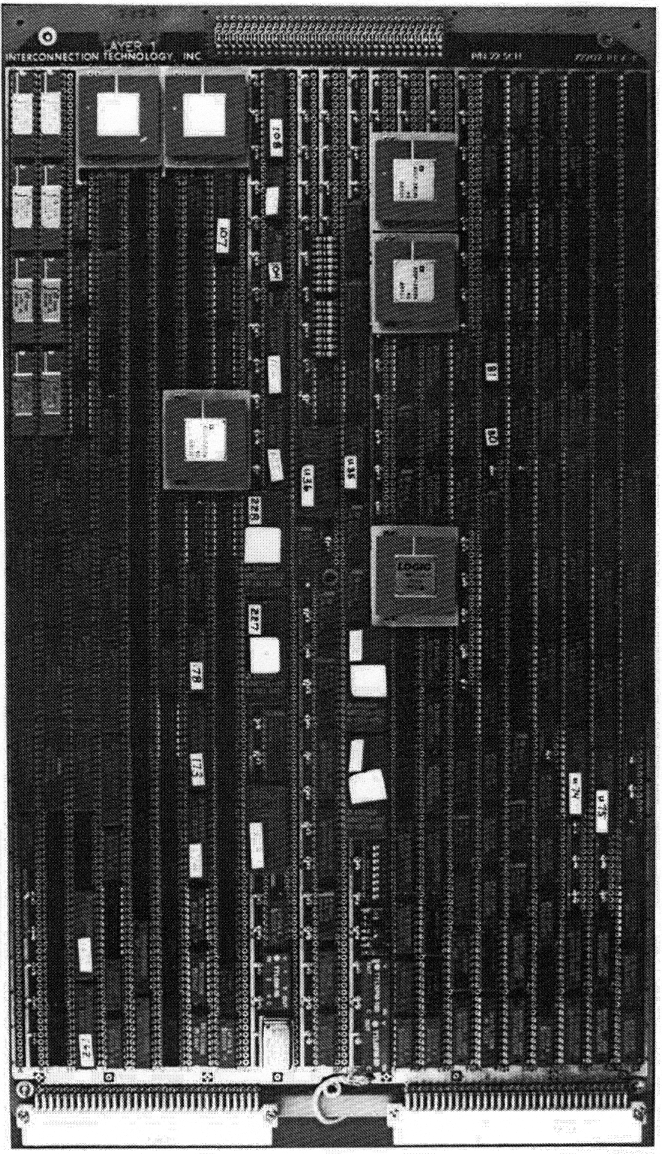
By inspection it would appear that the proper value of log2(fsΔf) = log2N is 8, 9, or 10. Assuming a nominal value of 9, we can use [link] to accurately estimate the optimum value of f_s. Performing this calculation yields 437 kHz. In the actual design, this value was rounded up to 512 kHz, the next-higher power-of-two integer multiple of 4 kHz. The choice of fs = 512 kHz in turn means that the tuner decimation M_t must equal 4 and the tuner's pulse response duration L_t must equal at least 20.

The resulting tuner/transmultiplexer, shown in [link] and described in [link], was built on a single circuit card. The 12-bit A/D module was mounted separately in the chassis. One multiplier chip operating at 4.096 megamultiplies/sec performed the tuner's quadrature downconversion. Two multiplier-accumulators (MACs) filtered and decimated the downconverted signal, preserving the center 248 kHz. Two more MACs perform the window-and-fold preprocessing for the transmultiplexer while a single MAC is used to compute the radix-2 FFT. Seven stages are used to compute the 128-point FFT and an additional one is used to perform sideband inversion on those voice channels designated by the user. This transmultiplexer also happens to use the so-called *offset-bin* DFT instead of the usual DFT. The motivation for this and the method for implementing it are discussed in Offset Bin Operation from "An Introduction to the FDM-TDM Digital Transmultiplexer: Appendix B".

Design of the FSK VFT Telegraphy Demodulator

The section "Example: Using an FDM-TDM Transmux to Demodulate R.35 Telgraphy Signals" discussed the use of an FDM-to-TDM transmultiplexer as an integral part of a demodulator capable of handling all 24 FSK signals present in an FDM voice frequency telegraphy (VFT) system. The analysis developed in that section showed that, in absence of other system-level factors, the best input sampling rate to the transmux-based filter bank was 3840 Hz, 64 times the 60 Hz fundamental tone spacing in the R.35 standard. In this section, we re-examine that choice in terms of the tuner required to provide the VFT signal to the transmultiplexer.

To pass all 24 FSK components of an R.35 VFT signal, the tuner must have a passband B_t of slightly more than 2880 Hz. The system must be able to accept real-valued digital samples from a commercial PCM link. These are provided at a rate of 8000 samples/sec[footnote]. From the section "Example: Using an FDM-TDM Transmux to Demodulate R.35 Telegraphy Signals" we recall that the other key parameters in the filter bank's design are: Q=3, M=12 (assuming the input rate is 3840 Hz), K=163, and N=64. Using the values in [link], and assuming a nominal value of 2.5 for α_t, yields 3920 Hz as the optimal value of f_s. This is very close to the best choice without taking the tuner into account. We therefore fix on 3840 Hz as the overall best choice.



990679

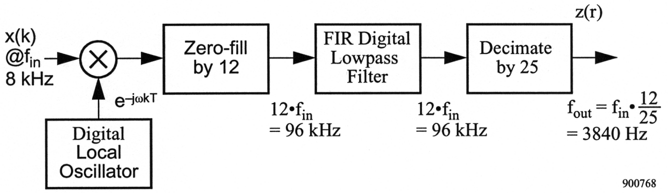
The next problem encountered, however, is that the choice of fin as 8000 Hz and f_s as 3840 Hz means that the tuner's decimation factor M_t is not an integer. In particular, with these sampling rate choices, M_t is given by 2512. As a result, a simple one-step decimating tuner of the type shown in Figure 1 from "Derivation of the equations for a Basic FDM-TDM Transmux" cannot be used directly.

The solution to this problem comes with the use of digital interpolation and decimation techniques. These are described in [link] and we refer to it here as digital resampling, the process of creating new digital samples at the desired rate from a sequence sampled at a different rate. The block diagram of this process is shown in the top portion of [link]. The incoming real-valued signal is first quadrature downconverted to move the band of interest into the passband of the digital lowpass filter and to register the filter bank's filters with the mark and space frequencies of the VFT signal. Conceptually, the downconverted quadrature signal is then zero-filled[footnote] by a factor of 12, lowpass filtered, and then decimated by a factor of 25. The zero-filling artificially increases the sampling rate to 96 kHz, creating 11 extra images of the input signal in the process. The lowpass filter removes these images and bandlimits the zero-filled signal to just the 2880 Hz band of interest. The decimation leads to an output rate of $9600025 = 3840$ Hz, exactly the desired value. In fact, the signal is never physically zero-filled. Pointers in the hardware keep track of where the non-zero data points lie and use that information to avoid doing unnecessary multiplication.

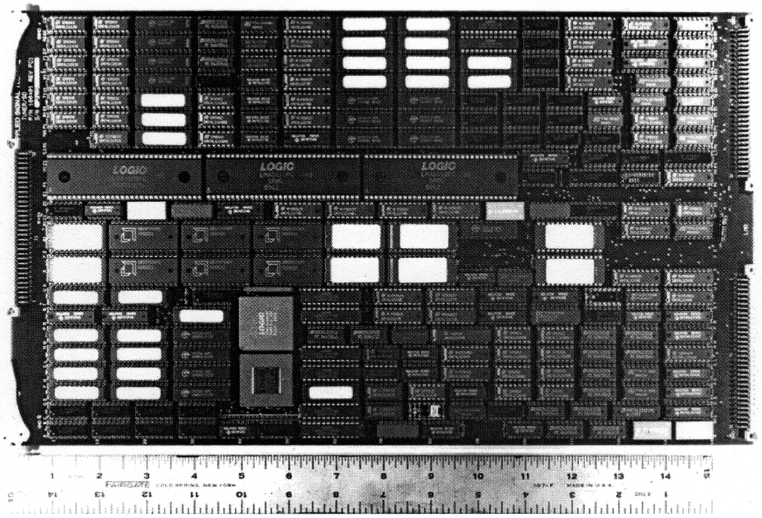
The bottom portion of [link] shows a circuit card assembly built to perform the downconversion and resampling processes for 24 input voice channels. An multiplier chip was used for the downconversion of all 24 channels and a pair of MACs performs the filtering needed to resample all 24 inputs. Programmable ROMs were used to generate the sequencing signals needed for the resampler. The extra MAC and ALU visible on the card are used to spectrum-analyze all 24 input channels with 60 Hz resolution at about 40 times a second. This spectral data is D/A-converted and provided to an oscilloscope for use by the equipment's operator.

Note that even though resampling is being performed, the equations used to choose the optimum value of f_s are still valid. The fundamental reason for this is that the filter segment of the tuner is still of the FIR variety and that one-step decimation is still employed. As a result, the average computation for the tuner remains as predicted by [link].

We might note in passing that the resampler used here is termed a *synchronous resampler* since the ratio of the number of input samples to output samples is rigidly fixed. It is also possible to employ a so-called *asynchronous resampler* to produce the desired samples. This is usually done when the input sampling rate varies slightly over time and it is desired to have the output rate locked to some frequency standard. The control of such resamplers is more complicated than the synchronous variety but the amount of computation needed for the downconversion and filtering is essentially the same.


x(k)
@f_{in}
8 kHz
$e^{-j\omega kT}$
Digital Local Oscillator
Zero-fill by 12
$12 \cdot f_{in}$ = 96 kHz
FIR Digital Lowpass Filter
$12 \cdot f_{in}$ = 96 kHz
Decimate by 25
z(r)
$f_{out} = f_{in} \cdot \frac{12}{25}$ = 3840 Hz
900768

a) Block Diagram of a Synchronous Digital Resampler



b) A Tuner Card, Which Performs Digital Resampling on 24 Simultaneous Inputs

ASIC-based Implementation of FDM Group Tuning and Transmultiplexing

A number of apparently inoffensive assumptions were made in the development of the tradeoff formulas used in the previous examples. One was that one-step (also called *single stage*) decimation is used in the tuner's filtering and the other is that the number of multiplications and additions forms good basis for comparing the complexity of various designs. This example demonstrates some counterexamples along the way to the description of a system that represents the current state of the art (circa 1990) in tuner and transmultiplexer design.

Suppose that our goal is to accept a full 2700-channel FDM telephone baseband, select an FDM group with a tuner, and then demultiplex the constituent 12 voice grade channels with a transmultiplexer. In the now-familiar way, we develop the certain specifications for the transmultiplexer and tuner separately and then jointly optimize the shared parameters.

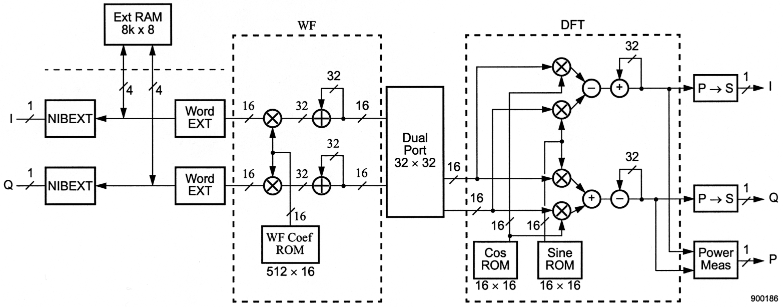
- Transmultiplexer: To achieve the desired channel shaping, we select Q to be 16. To minimize the amount of computation, we set K to unity. The window/tuner pulse response chosen provides an adjacent channel rejection of better than 55 dB and an NPR of about 55 dB.
- Tuner: A 2700-channel baseband extends up to 12388 kHz. Leaving a transition band for an analog antialias filter and looking for a power

of two times 4 kHz leads to the selection of 32768 kHz as fin, the baseband digitization rate. The tuner output bandwidth B_t must be at least 48 kHz to pass an FDM group. Owing the high tuner decimation required, we assume that α_t must be on the order of 3.

We now turn to [link] to determine the optimum value of f_s, and with it, Mt,Lt, and N. Plugging in to this equation yields an optimal f_s of about 490 kHz, more than ten times greater than the FDM group's bandwidth. In analyzing this result, we find that the amount of computation needed by a single-step FIR decimating tuner is so high that it dominates that needed by the transmultiplexer. Clearly another approach is needed.

In response to this problem, the company developed a pair of custom application-specific ICs (ASICs) for selecting FDM groups from digitized basebands and another chip for transmultiplexing four FDM groups. The block diagram is essentially the same as that shown in Figure 1 from "Derivation of the equations for a Basic FDM-TDM Transmux" except that a multistage decimating filter is used. In all, nine filter stages are employed. Each bandlimits the incoming signal sufficiently that a decimation by two is possible. The first few stages, the ones that must operate at very high rates, use pulse responses so simple (for example, h(k) = [1,2,1]) that only shifting and addition are needed. The effect of nine divisions by 2 is the reduction of the sampling rate f_s to 64 kHz. The 48-kHz-wide FDM group is thus represented at the output of the tuner chips as complex-valued samples at a rate of 64 kHz.

The transmultiplexer ASIC accepted four FDM groups, each quadrature-sampled at 64 kHz, and demultiplexes all 48 voice grade channels. A block diagram of a single path through the device is shown in [link]. The window-and-fold circuit was implemented by using onboard weighting coefficients and serial multipliers. The partial sums were stored in off-chip RAM. The output of the window-and-fold circuit was then transformed using a 16-point DFT. The complex-valued bin outputs, produced at a 4 kHz rate, were sent out over a serial interface.



Several of the design choices made with these chips are different that those seen earlier in the technical note. The first, seen in the tuner chips, is the use of multistage decimation. As [link] shows, this can almost always reduce the total amount of multiply-add computation needed for the tuner, at a certain cost in design simplicity. The other issue, evident in the design of both the tuner and the transmultiplexer, is that memory and control are at least as costly commodities in an ASIC design as are multiplications and additions. A vivid example is that the transmux ASIC used direct computation of the DFT rather than using an FFT. Even though the amount of multiplication is on the order of four times as much using the DFT, the overall DFT design used less silicon than the equivalent FFT.

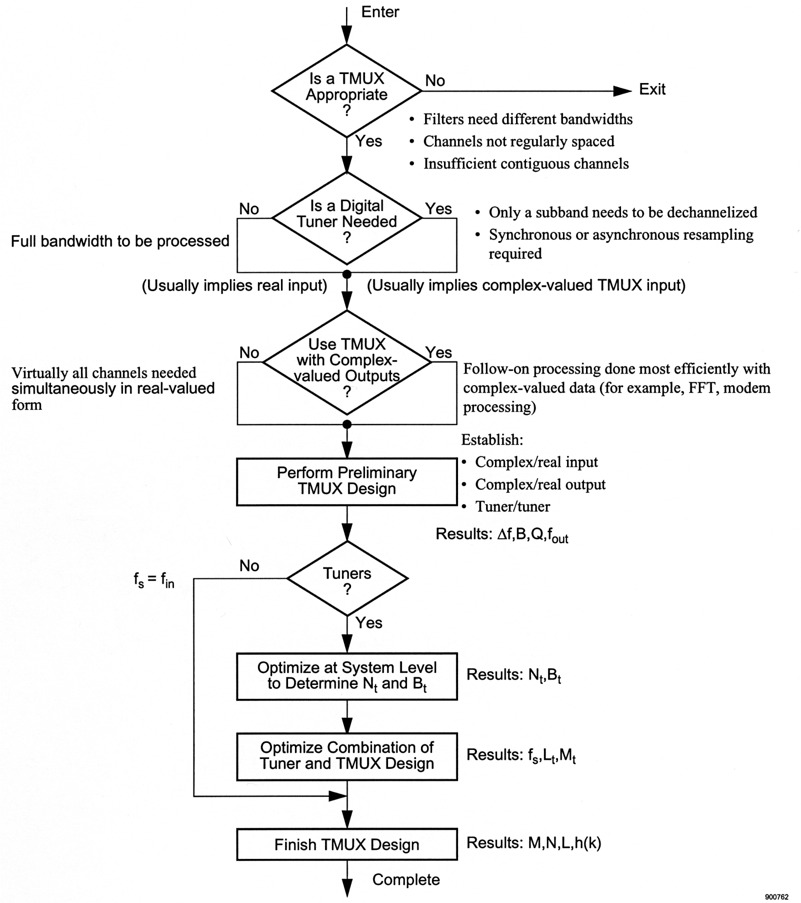
General Design Principles

There is a very large number of considerations that affect the selection of the best method of frequency-division demultiplexing signals in a particular application. As a result, it is virtually impossible to provide a simple *cookbook* methodology that always produces the best design. Even so, it is useful to systematically describe the design issues and choices evaluated so far in this technical note. Such a description, condensed into a design flowchart, is discussed in this section. Comparison of it with the design examples provided in the section "Example: Using an FDM-TDM Transmux to Demodulate R.35 Telegraphy Signals" and the section "The Impact of Digital Tuning on the Overall design of an FDM-TDM Transmux" shows excellent agreement. But while it is intended to be helpful, it must be used with care since relatively small differences in the application-dependent assumptions can influence the resulting choices quite considerably.

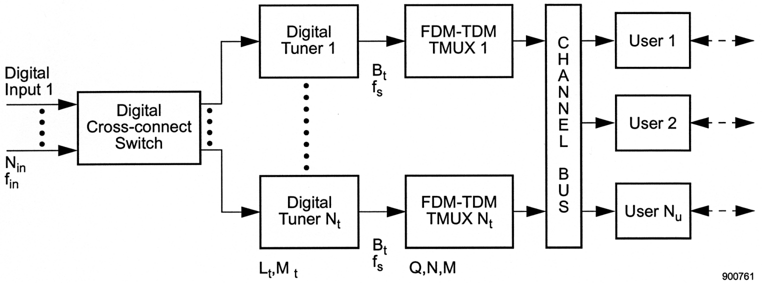
The decision flowchart presented in [link] assumes that the generalized demultiplexer has the block diagram shown in [link]. The system accepts Nin digitized FDM signals, all sampled at fin Hz. These are made available to N_t digital tuners. All of these tuners are of the same design, employ the same decimation factor M_p and produce output samples at the same rate of f_s Hz. The tuner outputs are transmultiplexed, sending their output channel samples to a bus, which up to N_u user processes have access to. Since each transmux is fed by a tuner, there are N_t transmuxes, each parameterized by Q,N, and M.

On the one hand, this architecture is not perfectly general, since parameters such as filter bandwidths are assumed to be identical, but it is representative of a very complex transmultiplexer-based system. On the other hand, it can be simplified considerably, by allowing Nin or N_t to be unity for instance, and still be reasonably described by the flowchart.

Flowchart for Determining the Applicability of Transmultiplexing to a Frequency-Division Demultiplexing Problem


Enter
Is a TMUX Appropriate ?
No
Exit
• Filters need different bandwidths
• Channels not regularly spaced
• Insufficient contiguous channels
Yes
Is a Digital Tuner Needed ?
No
Yes
Full bandwidth to be processed
• Only a subband needs to be dechannelized
• Synchronous or asynchronous resampling required
(Usually implies real input)
(Usually implies complex-valued TMUX input)
Use TMUX with Complex-valued Outputs ?
No
Yes
Virtually all channels needed simultaneously in real-valued form
Follow-on processing done most efficiently with complex-valued data (for example, FFT, modem processing)
Perform Preliminary TMUX Design
Establish:
• Complex/real input
• Complex/real output
• Tuner/tuner
Results: $\Delta f, B, Q, f_{out}$
Tuners ?
No
$f_s = f_{in}$
Yes
Optimize at System Level to Determine N_t and B_t
Results: N_t, B_t
Optimize Combination of Tuner and TMUX Design
Results: f_s, L_t, M_t
Finish TMUX Design
Results: M,N,L,h(k)
Complete
900762

Generic Block Diagram of an FDM Demultiplexer Requiring Digital Input Switching, Tuning, and Demultiplexing



The flowchart is shown in [link]. While perhaps self-explanatory, some commentary is provided for the faint-hearted.

- The first step is to determine whether an FDM-TDM transmultiplexer is really needed for the application. Generalizing wildly, a transmultiplexer is the right choice if three conditions are met:

    1. It is desired to simultaneously demultiplex a reasonably large (for example, 10 or more) number of contiguous channels from an FDM signal
    2. They are regularly spaced in frequency
    3. The same filter can be used for all of them without harm to the signals

  If these conditions aren't met, then altemative schemes, such as separate tuners for the desired channels, should be considered.
- Once it is determined that a transmultiplexer is needed, the next question is whether some form of digital tuner is needed to precede it. As a rule, no tuner is needed if:

    1. It is desired to demultiplex all of the channels seen in the full bandwidth of the input
    2. The input signal is sampled at a suitable rate

If resampling is needed, or if only a subband of the input signal's bandwidth is to be dechannelized, then a tuner is called for. Usually the use of a digital tuner leads to the use of a transmultiplexer that accepts complex-valued data while the absence of a tuner implies the use of a transmux that accepts real-valued data.

- The last major question is whether the outputs of the transmultiplexer should be real- or complex-valued. This usually depends completely on the processes using the transmultiplexer outputs. In some cases, such as commercial telephony (see the example in Appendix C), the outputs are desired to be in real-valued form so that they can be switched or formatted for TDM/PCM transmission. In other applications, however, particularly those that involve signal processing (for example, spectrum analysis), the use of complex-valued outputs is desired.
- With these fundamental system-level questions answered, the preliminary design of the transmultiplexer itself can begin. Based on the channel spacing, the desired filter frequency response, and the nature of the follow-on processing, such parameters as Δf,B,Q, and fout can be determined by using the rules presented in the section "Derivation of the equations for a Basic FDM-TDM Transmux".
- If no tuners are needed, then the design of the

> transmux can be completed by determining M,N,L, and the pulse response h(k). If tuners are needed, then the tradeoffs between the tuner and transmultiplexer design must be performed in order to know enough to finish the design of the transmux itself. The first step in this tradeoff is to determine the number of tuners N_t and their bandwidths B_t. The second step, given B_t, is the tradeoff identified in Section 5, which leads to the choice of the transmux input sampling rate f_s, and hence L_t and M_t.

Of these two steps, the first is often the more difficult since the optimization may be based on non-mathematical considerations. An example of this is the case in which a large number of contiguous FDM channels need to be demultiplexed from an even larger input band. Should there be a few tuners of large bandwidth or more with narrower bandwidth? A purely mathematical optimization using an objective function such as the number of multiply-adds will conclude that the former is better, while a user might prefer the selectivity (for example, *cherry picking*) afforded by a multitude of narrower tuners.

Conclusion

The FDM-TDM digital transmultiplexer has become an important building block in a variety of signal processors. The main goal of this technical note is to explain how the equations for it are derived and to provide some information about how its parameters are chosen in practical application.

In addition to fulfilling this role, additional information has been included about the design tradeoffs that must occur between the transmultiplexer and the processing steps preceding and succeeding it. In particular, we focused on the computational tradeoffs between the transmultiplexer and the digital tuner which frequently precedes it.

Several appendices are attached that describe variations of the basic FDM-TDM transmux, such as the offset-bin transmux, and related design issues, such as some of the considerations involved in the design of the weighting function h(k).

A final warning is in order. This technical note focuses on a fairly narrow topic, that is, radix-2 and radix-4 FFT-based, single-stage, FIR FDM-TDM transmultiplexers. While this subclass of transmultiplexers is widely used, it is by no means the only one. The referenced paper by Schuermann and Göckler [link] presents a broad overview of

FDM-TDM and TDM- FDM transmultiplexers and should be consulted by the serious engineer or student.

Appendix A

The section Derivation of the equations for a Basic FDM-TDM Transmux showed the FDM-TDM transmultiplexer can be viewed as an efficient implementation of a bank of digital tuners, and that the data-weighting function h(k) is just the pulse response of the FIR lowpass filter used in these equivalent tuners. We therefore approach the design of h(k) by designing the proper tuner pulse response.

The perfect filter pulse response would pass the signal of interest with no gain or phase distortion, would completely suppress all other FDM channels, and would require little computation. These are not all simultaneously achievable, of course, and the design of the actual filter is a compromise between these issues. It is further complicated by the fact that software packages are not generally available to perform some of the types of optimization needed to design these filters. We proceed first by examining how an optimal equal-ripple linear phase FIR filter performs.

Frequency Responses of Perfect and Realizable Tuner Filters Overlay of the Required Tuner Filter with the Generalized Response of an Optimal Linear Phase Equal-Ripple FIR Filter

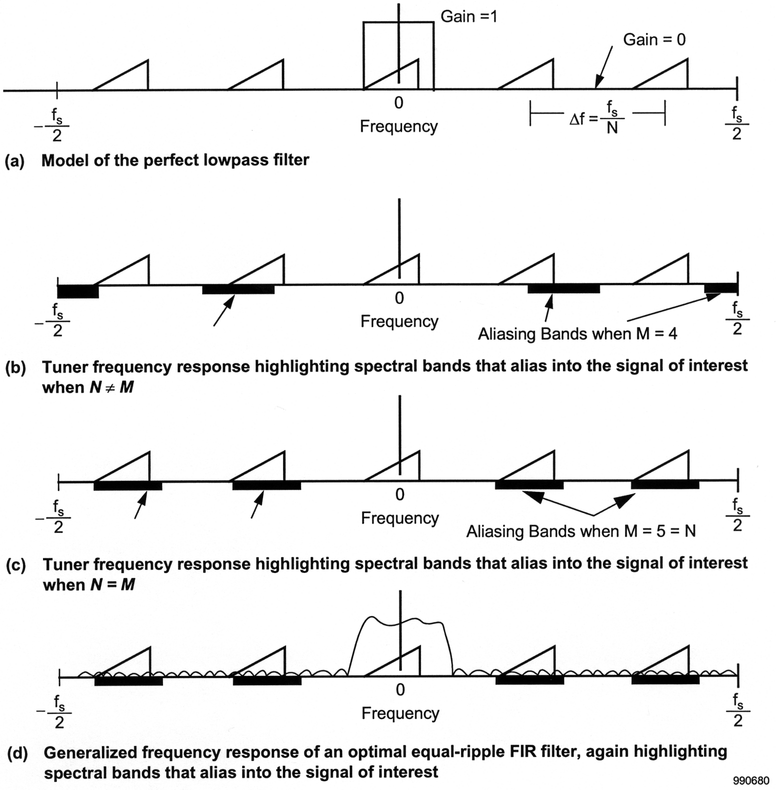
Use of Optimal, Linear Phase, Equal-

Ripple Design Techniques

The filter design problem at hand can be understood by examining [link]. The perfect filter, shown in [link](a), passes the channel of interest with unity gain and zero phase shift across its bandwidth of *B* Hz, centered at DC, and completely attenuates energy at all other frequencies between -fs2 and fs2.

In fact, it is not necessary to suppress all out-of-band energy to protect the signal of interest. The principal reason for this filtering is to suppress the out-of-band components that alias into the band of interest when the output of the tuner (that is, transmultiplexer) is decimated by the factor *M*. These bands are shown in [link](b) for the general case in which $M \neq N$, while [link](c) shows the important special case of the *basic FDM-TDM transmux* in which $N = M$. In the latter case, the FDM channels not of interest alias directly onto the signal of interest while, in the former, the channels not of interest may be spread around the band more.

As alluded to earlier, practical FIR filters of finite duration cannot pass the signal interest perfectly and suppress all other energy completely. The response shown in [link](d) is the generalized response of a good practical approximation, the response provided by an optimal FIR linear-phase, equal-ripple filter of the sort designed by the Parks-McClellan software package. These filters provide flat differential group delay and allow the designer to optimally trade between passband ripple, stopband suppression, and transition band as a function of the filter order *L*. A description of this general filter design methodology can be found in [link].


Gain =1
Gain = 0
$-\frac{f_s}{2}$
0
Frequency
$\Delta f = \frac{f_s}{N}$
$\frac{f_s}{2}$
Aliasing Bands when M = 4
Aliasing Bands when M = 5 = N
990680

(a) **Model of the perfect lowpass filter**

(b) **Tuner frequency response highlighting spectral bands that alias into the signal of interest when *N* ≠ *M***

(c) **Tuner frequency response highlighting spectral bands that alias into the signal of interest when *N* = *M***

(d) **Generalized frequency response of an optimal equal-ripple FIR filter, again highlighting spectral bands that alias into the signal of interest**

Superimposing [link](c) with the optimal response shown in Figure 1 of [link] to produce [link] shows that we must specify the channel bandwidth *B*, the transition bandwidth δf, the input sampling rate f_s, the degree of passband ripple tolerable, denoted *PBR* and the minimum tolerable stopband attenuation in dB, denoted *SBR*. A multi-term empirical formula can be found in [link] which determines the filter *L* quite accurately for a given set of design parameters. Reference [link] simplifies the Rabiner and Gold formula considerably to produce the design equation

$$L = \alpha f s \delta f$$

where L,fs, and δf are as just defined, and *α* is given by

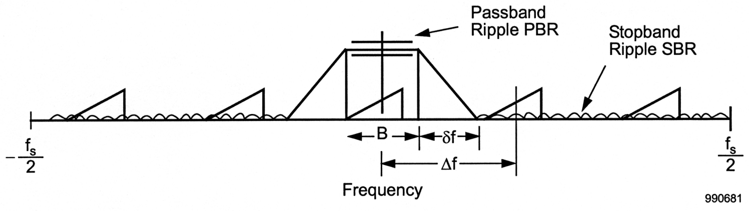
$$\alpha = 0.22 + 0.0366 \cdot S B R$$

The validity of these simplified formulas depends on a number of assumptions, detailed in [link], but all of them are sufficiently satisfied in this case to permit accuracy in the estimation of *L* within 5% or so.

Examination of [link] shows that δf, the filter transition band, can be no larger than Δf-B, the difference between the channel spacing and the bandwidth of each channel. Recalling also that N·Δf = fs, we find that

$$L = N \alpha \Delta f \delta f = N \alpha 1 1 - B \Delta f .$$

Thus, to first order, the pulse response duration of the required filter is proportional to the number of channels *N* and is hyperbolic in the *percentage bandwidth*, the ratio of the channel bandwidth *B* to the channel spacing Δf. The effect of the proportionality to *α* will be examined shortly.



Relationship to the Design Parameter *Q*

The development presented in the section Derivation of the equations for a Basic FDM-TDM Transmux defined the integer variable *Q* as the ratio of *L* and *N*. It was pointed out there without proof that in fact *Q* was an important design parameter, not just the artifact of two others. This can now be seen by combining the relationship $L \equiv QN$ with [link] to produce an expression for *Q*:

$$Q = \alpha \Delta f \delta f = \alpha 1 1 - B \Delta f$$

Since *N* depends strictly on the number of channels into which the input band is divided, *Q* contains all of the information about the impact of the desired filter characteristics.

Continuation of the Telegraphy Demodulation Example

Consider again the example of demodulating R.35 FDM FSK VFT canals discussed in the section

Example: Using an FDM-TDM Transmux to Demodulate R.35 Telegraphy Signals. In that section, we determined that the following parameters would be appropriate: fs = 3840 Hz, N = 64, and Δf = 60 Hz. To determine *Q*, and hence the rate of computation needed for the data weighting segment of the transmultiplexer, we need to specify *B* and SBR, the degree of stopband suppression required.

Generally speaking, the filters in an FSK demodulator need to have unity gain at the mark or space frequency and zero gain at the space or mark frequency, respectively. A computer simulation used to verify the design of the demodulator showed that suppression of 50 dB was more than enough to provide the needed performance. At first glance it might appear that the transition band δf can be allowed to equal the tone spacing Δf = 60 Hz, making the percentage bandwidth equal to zero. Actual FSK VFT systems, however, sometimes experience bulk frequency shifts of several Hertz. In order to maintain full performance in the presence of such frequency offsets, the tuner filters need to be designed with a passband bandwidth of 15 Hz or so. Using SBR = 50 dB in [link], we find with [link] that the required value of *Q* for this application is about 2.71. The actual value chosen for this application was 3, producing a pulse response duration of L = QN = 192, with the remaining degrees of freedom in the filter design used to widen the filter still more, allowing for even more frequency offset. For a 960-channel transmultiplexer this extra suppression goes up to 29.8 dB Filter Design Alternatives to Reduce the Required Filter Order

Implications of the Filter Design on Signal-to-Noise Ratio and Noise Power Ratio

The pulse response h(k) chosen for the transmultiplexer determines many of the transmux's key technical performance parameters, including:

- passband bandwidth, gain, and gain ripple
- passband differential group delay (constrained to zero by using an FIR linear phase design approach)
- adjacent channel rejection (also known as *intelligible crosstalk*)
- channel signal-to-noise ratio (SNR) (also known as *unintelligible crosstalk* and noise power ratio (NPR), a name based on one method of measurement)

We focus here on the last two. At first it might seem that these two are equivalent, but in fact the second is usually the more demanding of the two. This may be demonstrated by re-examining [link](a). An adjacent channel rejection specification of 55 dB, say, means that no signal appearing anywhere in the input bandwidth of f_s Hz can appear in the band of interest at any level higher than 55 dB below the signal of interest. If the signal of interest and the one not of interest have the same power levels, then this specification implies that the filter pulse response should suppress the signal not of interest by at least 55 dB before it is potentially aliased into the band of interest. Thus the adjacent channel rejection specification treats each interferer separately and forces each of them to be suppressed to a level below intelligibility. Typical specifications for voice channel demultiplexers, for example, are 55 dB of suppression for any signal more than 300 Hz above or below the channel of interest.

The last specification limits the total noise that enters the band of interest from other channels. These channels are assumed to be statistically independent, implying that whatever energy that aliases into the band of interest from each of the channels is uncorrelated with the others, that their powers add, and that none of them is individually intelligible.

The channel SNR can be quantified by using the expression

$$S N R = \int - \pi B \pi B H (\omega) 2 P c (\omega) d \omega \Sigma n = 0 , n \neq c C - 1 \int 2 \pi n \Delta f - \pi B 2 \pi n \Delta f + \pi B H (\omega) 2 P n (\omega) d \omega$$

where *B* is the bandwidth of the channel of interest, *C* is the number of channels with signals present, *c* is the index of the band of current interest, H(ω) is the frequency response of the pulse response h(k), and Pn(a)) is the power spectrum of the *n*-th signal.

If we assume that all *C* channels have the same average power *P*, that the power gain of the filter is unity for the channel of interest, and that all other channels are suppressed by exactly *S* dB, then [link] simplifies to

$$SNR = S - 10 \cdot \log 10 (C - 1) \text{ in dB}$$

The implications of [link] can be seen with an example. Suppose that the application at hand requires the demultiplexing of 60 FDM voice channels, that the adjacent channel rejection is specified to be 60 dB, and that the unintelligible crosstalk or SNR specification is 55 dB. Evaluation of [link] shows that *α*, on which *Q* and *L* depend, needs to be about 2.42 to suppress any single component by 60 dB. [link], however, shows that to satisfy the unintelligible noise specification, all channels not of interest need to be suppressed by an extra $10 \cdot \log 1059 = 17.8$ dB[footnote]. To meet this specification, the pulse response needs to suppress the other input channels by $55 + 17.8 = 72.8$ dB, with the associated growth in *α* from 2.42 to 2.88 and some additional design concern in hardware using finite word length arithmetic.

Combining [link], [link], and [link], we find that *L* is given by

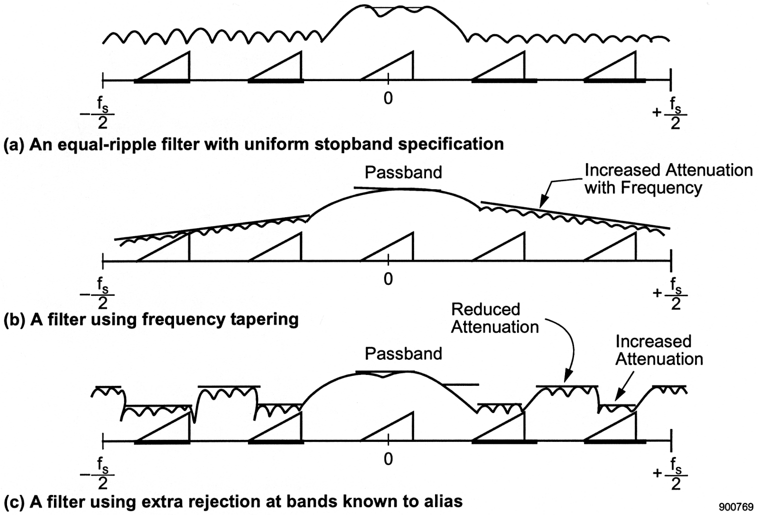
$$L = N \alpha (C) \Delta f \delta f = N \alpha (C) 1 1 - \Delta f B$$

where α(C) is given by

$$\alpha (C) = 0.22 + 0.0366 (S N R r) + 0.366 \log 10 (C - 1)$$

and SNRr is the required SNR performance in dB. Note that when *C* is a large fraction of *N*, which is usually the case, the pulse response duration *L* actually grows faster than proportionally to the number of bins *N*.

A warning is in order here. While accurate in principle and generally accurate numerically, this section presents a simplified view of the filter optimization problem and the implications of each of the technical requirements. Each actual application requires a careful evaluation of the specifications appropriate to it and the impact to each of the transmultiplexer's design parameters. In addition, note that we used [link] to reach some of these conclusions when, in fact, [link] is really the right one to use. To illustrate how this might affect the resulting design, observe that [link] implicitly assumes a pulse response of the type shown in [link], which suppresses all channels not of interest about equally. Consider then the frequency responses shown in [link]. The first is a standard Parks-McClellan design in which the stopband ripple objective is the same over the whole stopband. The second two are alternative designs that use similar or less amounts of computation. The one shown in [link](b) slowly increases the stopband suppression with higher frequency, essentially removing those channels from the SNR calculation. Another scheme, shown in [link](c), obtains added suppression in the bands known to alias into the band of interest by releasing control in the bands that will not alias in. In passing, it should be noted that the Parks-McClellan software package can be modified to perform both of these filter designs. To summarize, the equations presented in this section serve as a good guide to the selection of the pulse response h(k) and its duration *L*, but skillful use of [link] and the full design formulas for FIR linear phase filters can reduce *L* and the implied required real-time computation level by 10 to 40%.



(a) An equal-ripple filter with uniform stopband specification

(b) A filter using frequency tapering

(c) A filter using extra rejection at bands known to alias

Example of a Voice Channel Demultiplexer

Several of the examples presented in The Impact of Digital Tuning on the Overall design of an FDM-TDM Transmux concern the use of FDM-TDM transmultiplexers to demultiplex the voice channels found in an FDM telephone baseband. In this section, we examine briefly an example of the design of such a transmultiplexer. For the purpose of this example, we focus on the design of the pulse responses for the group transmultiplexer VLSI chip shown in Figure 6 from The Impact of Digital Tuning on the Overall design of an FDM-TDM Transmux.

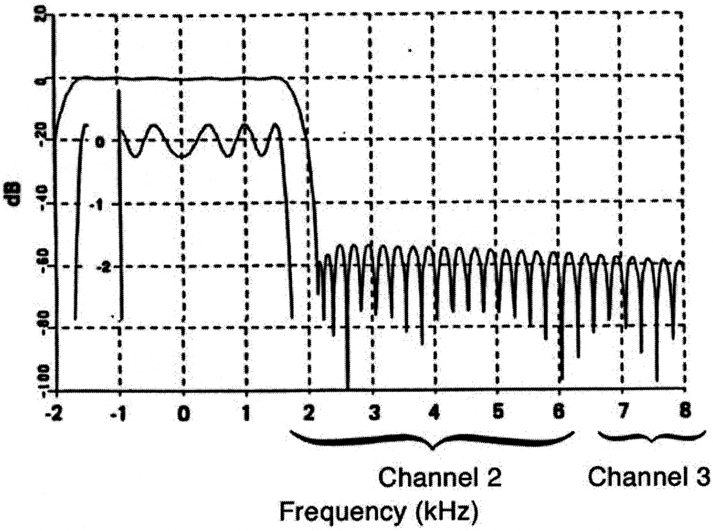
Repeating from The Impact of Digital Tuning on the Overall design of an FDM-TDM Transmux, the general design parameters for the group transmultiplexer are: $fs = 64$ kHz, $N = 16$, $\Delta f = 4$ kHz, $C = 12$, and the 3-dB bandwidth $B = 3700$ Hz. We desire the passband to be as flat as possible, that the adjacent channel rejection meet or exceed 55 dB at 300 Hz into adjacent channels, and that the SNR and NPR meet or exceed 52 dB. We also strongly desire that Q equal 16, since such a power-of-two value would simplify the design of the hardware.

We do a first cut by evaluating δf to be 450 Hz, the difference between the edge of the equal-ripple passband and the point 300 Hz into the adjacent channel. Suppose that we optimistically assume that only 55 dB of suppression is needed in the stopband. Using these values, plus the fact that fs = 64 kHz, in [link] yields L = 318, which implies a value of Q = 19.85. This is close to, but exceeds, a *nice* power of two, that is, 16. By working the filter design problem carefully it is possible to design pulse responses that do meet the requirements. Two of these are shown in [link]. One has a wide passband, at the expense of greater passband ripple, while the other trades bandwidth for ripple performance. The filter that was developed for this purpose used integral ROM to hold these pulse responses and allowed the user to control which is to be employed at a given time.

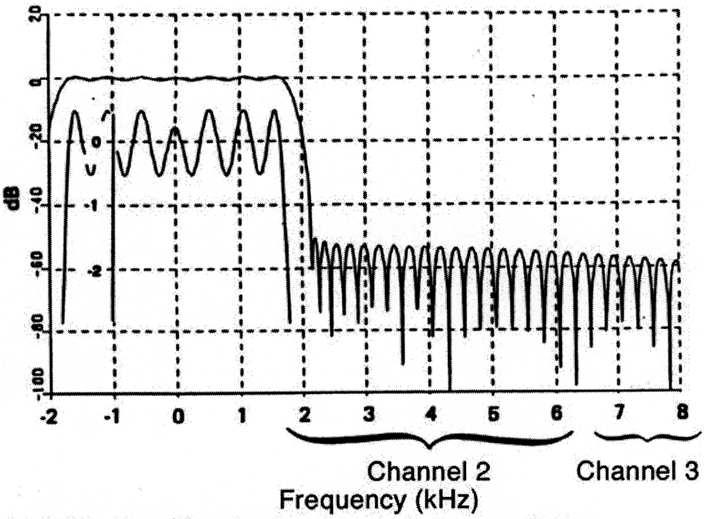
Frequency Response of FIR Filter Designed for a Voice Channel Transmultiplexer

Other Criteria for Filter Design

The focus of this section has been on the design of the transmultiplexer pulse response when viewed as a single tuner. In fact, most are designed this way. There are other applications however, that require that other considerations enter the design process. An example is the interference canceller discussed in A Pair of Examples from An Introduction to the FDM-TDM Digital Transmultiplexer: Appendix C. In this case, the filter pulse response is designed to bandlimit, as before, but in addition, constraints are introduced that have the effect of guaranteeing good broadband behavior as well.



a) Narrower Passband with Lower Passband Ripple



b) Wider Passband with Greater Passband Ripple

Appendix B

Spectral Implication of Each Step of the Sample-Rate Doubling Used to Produce Real-Valued OutputsWe use the generic notation h(k) and *L* as the pulse response and its duration just as we did in Section 3, although they are distinct. They are not completely independent, however, since the design of this filter is usually impacted by the design of the transmultiplexer weighting function. Block Diagram of the Processing Required to Produce Real-Valued Outputs with Complete Sideband Control for an Offset-Bin Input

Production of Real-valued Outputs

It is frequently the case that the best system-wide choice is the use of a transmultiplexer producing complex-valued outputs even though some of the system outputs need to be real-valued. An example of this is when a transmultiplexer is used to supply all transmuxed signals to an analyzer of some sort, plus a few selected signals for a downstream processor or transmission system. If the analysis processing is best done with complex-valued data (and it often is), then the best system-level choice is often to use a complex-output transmux and then to convert the relatively few system outputs to a real-valued representation. This appendix describes how this can be done.

Producing a real-valued signal from a complex-valued one is as simple as taking its real part, but for this to be valid, the complex-valued signal must be sampled frequently enough. In actual practice, this condition is not usually met and it is necessary to increase the complex-valued signal's sampling rate by a factor of two before extracting the real part. Another complication is that the user may desire to choose the spectral orientation of the resulting real-valued signal.

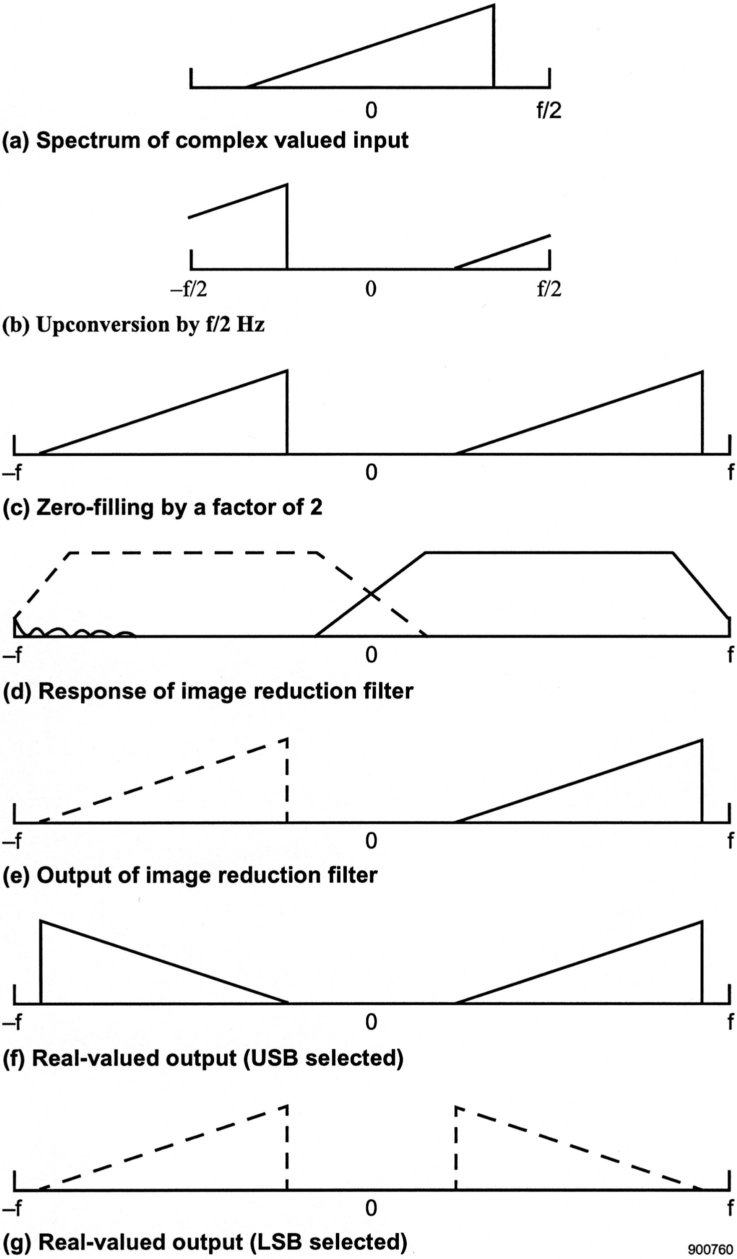
[link] shows the spectral implications of the steps usually taken to interpolate the complex-valued signal, specify its sideband orientation, and produce a real-valued representation.

[link](a) shows the assumed spectrum of the complex-valued input signal z(r). The signal z(r) is assumed to be the output of a complex-valued FDM-TDM transmultiplexer. It is sampled at the rate *f* and is spectrally oriented so that an increasing input frequency results in a higher output frequency. We'll term this *upper sideband*. Note that the bandwidth of z(r) is less than *f*.

We now upconvert z(r) by f2 Hz by multiplying it by (-1)r. This is shown in [link](b). The signal is now centered around f2 instead of 0 Hz. The sampling rate is doubled by zero-filling z¯(r), that is, z¯(k) is created by setting z¯(k = 2r) = z(r)(-l)r for even values of *k* and z¯(k) = 0 for odd values of *k*.

The spectral effects of this zero-filling are shown in [link](c). The sampling rate is now 2f and two upper sideband images of the original signal are present, one centered at f2 and the other at -f2.

The next step is to bandpass filter the desired image. Two filter transfer functions are shown in [link](d). We focus first on the one drawn with the solid line for now. When the zero-filled input signal Z¯(k) is convolved with the filter having this response, the higher of the two images is preserved and the lower one suppressed. This is shown in [link](e).



(a) Spectrum of complex valued input

(b) Upconversion by f/2 Hz

(c) Zero-filling by a factor of 2

(d) Response of image reduction filter

(e) Output of image reduction filter

(f) Real-valued output (USB selected)

(g) Real-valued output (LSB selected)

Obviously, the use of a filter with transmission characteristics shown in the dashed lines in [link](d) would have the effect of selecting the other image, the one shown in dashed lines in [link](e).

As a result of the filtering, only one image remains and which image it is depends on the bandpass filter chosen. The last step is the production of the real-valued output by simply taking the real part of the complex-valued signal that appears at the filter's output. Since real-valued signals must have spectral symmetry about the origin, this extraction of the real part has the effect of producing an another image of the input, only this one has the opposite sideband orientation. If the upper image is selected by the bandpass filter, then spectrum of the output signal is as shown in [link](f). We term this as *upper sideband* since the positive frequency image has the same orientation as the original complex-valued input signal. Choosing the lower image with the bandpass filter has the effect of producing the spectral relationship shown in [link](g), the so-called *lower sideband* or *inverted case*.

To summarize, we produce the real-valued signal by

1. Upconverting by f2
2. Zero-filling by a factor of 2
3. Bandpass filtering one of the two images created by the zero-filling
4. Taking the real part of the complex filter output

The sideband orientation of the output is determined by which image is selected by the filter. We now develop the equations that describe these processing steps.

We define the bandpass filter output to be $\bar{y}(k)$, given by

$$\bar{y}(k) = \Sigma\, l = 0\, L - 1\, \bar{z}(k - l)\, h(l)\, j\, l\, s$$

where the zero-filled input $\bar{z}(k)$ is as earlier defined. We recognize h(l)jls as the pulse response of the bandpass filter and *L* as its duration[footnote]. The factor *s* has the value of 1 or -1 to determine which of the two bandpass filters is desired, and, with it, which output sideband orientation is selected. The pulse response is written in this curious fashion to emphasize that h(l) is the real-valued pulse response of a lowpass filter. It is converted into the pulse response of a bandpass filter centered at ±f2 by multiplying it point by point by a sampled complex-valued sinusoid of frequency f2, if s=1, or -f2, if s=-1, that is, jls.

With no loss of generality we assume that *L* is an integer factor of two. Suppose now that *k* is even. If so, k=2r and

$$\bar{y}(k = 2r) = \Sigma\, u = 0\, L\, 2 - 1\, (r - u)(-1)\, r - u\, h(2u)\, j\, 2\, u\, s = (-1)\, r\, \Sigma\, u = 0\, L\, 2 - 1\, z(r - u)\, h(2u).$$

Now suppose that *k* is odd. If so, then it can be

represented as k = 2r + 1 and the expression for the output becomes
y ¯ (k = 2 r + 1) = Σ u = 0 L 2 - 1 z (r - u) (- 1) r - u h (2 u + 1) j (2 u + 1) s = (- 1) r Σ u = 0 L 2 - 1 z (r - u) h (2 u + 1) j s .

Using the assumption that the filter pulse response function h(k) is real-valued, then y(k), the desired real-valued output, is given
y (k) = R e { y ¯ (k) } = Σ u = 0 L 2 - 1 R e { z (r - u) } h (2 u) , k m o d 4 = 0 , = - s Σ u = 0 L 2 - 1 I m { z (r - u) } h (2 u + 1) , k m o d 4 = 1 , = - Σ u = 0 L 2 - 1 R e { z (r - u) } h (2 u) , k m o d 4 = 2 , = s Σ u = 0 L 2 - 2 I m { z (r - u) } h (2 u + 1) , k m o d 4 = 3 .

This set of equations can be written in a shorthand form by using vector notation. To do this we define X_k, Y_k, H_e, and H_o by the expressions
X k = 2 r = R e [z (r)] R e [z (r - 1)] … R e [z (r - L 2 + 1)] , Y k = 2 r + 1 = I m [z (r)] I m [z (r - 1)] … I m [z (r - L 2 + 1)] , H e = [h (0) h (2) h (4) … h (L - 2)] t , a n d H 0 = [h (1) h (3) h (5) … h (L - 1)] t ,

where the superscript *t* indicates the transpose of a vector. Using these definitions, the real-valued output y(k) can be written compactly as
y (k) = X k H e , k m o d 4 = 0 , y (k + 1) = - s Y k + 1 H 0 , k m o d 4 = 1 , y (k + 2) = - X k + 2 H e , k m o d 4 = 2 , y (k + 3) = s Y k + 3 H 0

$, k \bmod 4 = 3.$

Note that if FIR filters are employed, each output requires L2 multiply-adds. Thus the production of each real-valued output signal uses fL multiply-adds per second.

Now consider the case in which the signal of interest is supplied by an offset-bin transmultiplexer. If so, the initial multiplication by (-1)r is not needed. The effect of this can be seen by re-examining the equation for y¯(k) when *k* is even.

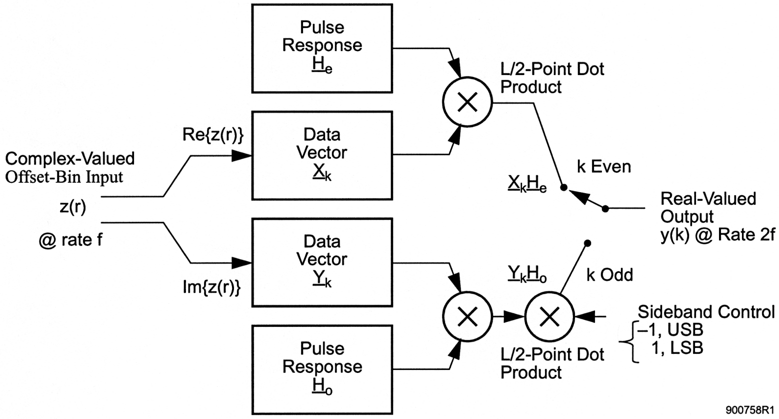
$$y^{-}(k=2r)=\Sigma u=0L2-1z(r-u)h(2u)j2su=\Sigma u=0L2-1z(r-u)h(2u)(-1)u=\Sigma u=0L2-1z(r-u)h^{-}(2u)$$

where h¯(2u) is defined by the equation h¯(2u) = h(2u)(-1)u,0 ≤ u ≤ L2-1.

Suppose that we similarly define h¯(2u + 1) by the expression h¯(2u + 1) = h(2u + 1)(-1)u, and then H¯e and H¯0 as in [link](fourth line). If we do this, we find that the expression for y(k) becomes simpler yet:

$$y(k:keven)=XkH^{-}e,\text{and}\,y(k:kodd)=-sYkH^{-}0$$

A block diagram of the processing needed to implement these equations appears in [link].

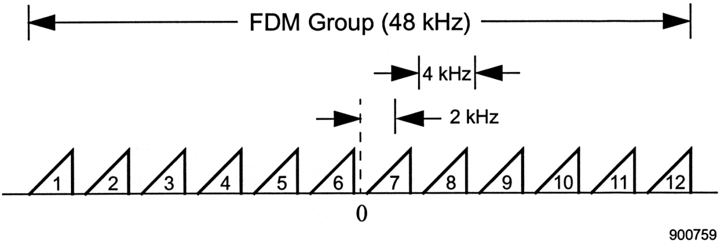


Use of an Offset-Bin Channel Bank to Separate the Channels in an FDM Group

Offset Bin Operation

The analysis presented to this point assumes that the tuning frequencies are integer multiples of some fundamental step size Δf. This implies that the 0-th_ bin or channel is centered at 0 Hz. While this is true in some applications, there are others in which the bin or channel centers are offset in frequency by Δf2 An example is shown in [link]. For this example, we suppose that an FDM group of twelve channels is digitally tuned and filtered, that is, it is quadrature downconverted so as to center the group at 0 Hz. ASICs such as those discussed in the section The Impact of Digital Tuning on the Overall design of an FDM-TDM Transmux can perform this function. [link] shows the group centered at DC, which places channels 1-6 below DC and channels 7-12 above. The channels are still separated by 4 kHz but their center frequencies are offset from DC by 2 kHz.

There are several solutions to this problem, the most obvious being to off-tune the tuner by 2 kHz. As this appendix will show, however, the FDM-TDM transmultiplexer equations can be easily modified to introduce the desired offsets.



To produce the desired set of equations, we have to repeat some of the formulation developed in Section 3. Frequency steps of Δf are still employed. The fundamental difference is that each tuner frequency is not an integer multiple of Δf but rather is a half-integer multiple, for example, $\omega = 2\pi(n+12)\Delta f$, where *n* is an integer. The effects of this substitution can be seen by joining the analysis in the section Derivation of the equations for a Basic FDM-TDM Transmux at [link] from Derivation of the equations for a Basic FDM-TDM Transmux. Substituting this new expression for the tuning frequency yields $y n(r) = e - j 2\pi(n + 1 2) r M N \cdot \Sigma l = 0 L - 1 h(l) x(r M - l) e j 2\pi(n + 1 2) l N$

As before, we subscript the decimate output y(r) by the parameter *n* but in this case it indicates that the tuning frequency is given by f0=(n+12)·Δf.

As before, we define the integer indices *q* and *p* by the expressions
l ≡ q N + p , where 0 ≤ q ≤ Q - 1 and 0 ≤ p ≤ N - 1

yielding
y n (r) = e - j 2 π (n + 1 2) r M N · Σ l = 0 L - 1 h (l) x (r M - l) e j 2 π (n + 1 2) l N = e - j 2 π n r M N e - j π r M N · Σ p = 0 N - 1 Σ q = 0 Q - 1 h (q N + p) x (r M - q N - p) e j 2 π (n + 1 2) (q N + p) N = e - j 2 π n r M N e - j π r M N · Σ p = 0 N - 1 Σ q = 0 Q - 1 h (q N + p) x (r M - q N - p) e j 2 π (n + 1 2) q N N e j 2 π (n + 1 2) p N = e - j 2 π n r M N e - j π r M N · Σ p = 0 N - 1 Σ q = 0 Q - 1 h (q N + p) x (r M - q N - p) (- 1) q · e j 2 π (n + 1 2) p N = e - j 2 π n r M N e - j π r M N · Σ p = 0 N - 1 e j 2 π (n + 1 2) p N Σ q = 0 Q - 1 h (q N + p) x (r M - q N - p) (- 1) q

Suppose we now define the variable v¯(r,p) by the expression
v ¯ (r , p) ≡ Σ q = 0 Q - 1 h ¯ (q N + p) · x (r M - q N - p)

and the pulse response h¯(qN+p) by the expression
h ¯ (q N + p) = (- 1) q · h (q N + p)

Substituting v¯(r,p) into the equation for the decimated output yn(r) of the tuner tuned to frequency f0=(n+12)·Δf yields

$$y_n(r) = e^{-j2\pi nrK} e^{-j\pi rK} \cdot \sum_{p=0}^{N-1} e^{j2\pi (n+\frac{1}{2}) \frac{p}{N}} \bar{v}(r,p)$$

In the section Derivation of the equations for a Basic FDM-TDM Transmux, we defined the *basic FDM-TDM transmultiplexer* as the special case of general transmultiplexer in which we set the decimation factor *M* to equal the number of channels *N*. This implies directly that K=1. This assumption leads to the corresponding *basic transmux* equations for the offset-bin case:

$$y_n(r) = (-1)^r \sum_{p=0}^{N-1} e^{j2\pi (n+\frac{1}{2}) \frac{p}{N}} \bar{v}(r,p) = IDFT_o\{\bar{v}(r,p)\} \text{ where } \bar{v}(r,p) \equiv \sum_{q=0}^{Q-1} h(qN+p)(-1)^q x((r-q)N-p)$$

and IDFTo{.} indicates the offset-bin inverse DFT.

While not immediately obvious, it can be shown that the offset-bin DFT can be computed with an FFT-like algorithm. A listing of one is shown in the following code. It results from a simple modification (that is, the initialization of *U*) in the FFT routine appearing on page 367 of [link].

FORTRAN Subroutine for an N-Point Offset-Bin FFT (Modified from DIT FFT)

```
SUBROUTINE OFFSET-FFT  (X, N, M)
C    OFFSET_FFT - computes the half-bin offs
C    decimation-in-frequency (DIF) FFT. The
C    and must have length N = 2**M.
```

```
C    The subroutine is entered with data in
C    exits with the DFT stored there.
C
     COMPLEX X(1), U, W, T
     NV2 =  N/2
     NM1 = N-1
     J = 1
C
     DO 7 I=1,NM1
     T = X(J)
     X(J) = X(I)
     X(I) = T
5    K = NV2
6    IF (K .GE. J) GO TO 7
     J = J - K
     K = K/2
     GO TO 6
7    J = J + K
C
     PI = 3.14159265358979
C
     DO 20 L=1,M
     LE = 2**L
     LE1 = LE/2
     U = CMPLX(COS(PI/FLOAT(LE)),SIN(PI/FLOA
     W = CMPLX(COS(PI/FLOAT(LE1)),SIN(PI/FLO
C
     DO 20 J=1, LE1
          DO 10 I=J,N,LE
               IP=I+LE
               T = X(IP)*U
               X(IP) = X(I) - T
10             X(I) = X(I) + T
20    U = U*W
C
          RETURN
     END
```

When an offset-bin transmux is performed, it is common not to premultiply by (-1)r as shown in [link]. This has the effect of frequency-converting the output signal by fout2=fs2M. When M=N (hence K=1), the spectral effect of this is as shown in [link](a) and (b). Instead of producing a complex signal centered at DC, not premultiplying by (-1)r centers the signal at fout2 In addition to obviating the need for a multiplication, this has the effect of moving the signal away from DC. This tends to improve signal quality since many finite-word length effects arising from hardware implementations (for example, truncation) produce spurious signals at DC.

A sequence has Hermitian symmetry if the real parts are symmetrical about the midpoint of the sequence while the imaginary parts are antisymmetrical.

Operation with Real-Valued Inputs

There are practical applications of FDM-TDM transmultiplexers in which the designer wants to extract all channels from a real-valued input. Such a signal can be applied directly to an FFT-based transmux of the variety described in the section Derivation of the equations for a Basic FDM-TDM Transmux but, since such a transmux is designed for use with complex-valued data, it might appear that unneeded computation is being performed. That is in fact the case. This section shows how the real-valued nature of the input can be exploited to reduce the required computation by slightly less than a factor of two.

Assume for this discussion that the input signal x(k) is real-valued and sampled at a rate of fs=2NΔf, where Δf, as before, is the frequency spacing between channels, and *N* is the maximum number of *unique* channels. The Nyquist theorem requires that f_s be twice NΔf since the input is real-valued. Half of the 2N channels present in the real-valued input are sideband reversed images of the other *N* channels. Thus we work to find an expression for those *N* unique channels. Using the basic equation for the FDM-TDM transmux (see equation 14 from Derivation of the equations for a Basic FDM-TDM Transmux), the *n*-th output is given by

$$y_n(r) = \sum_{p=0}^{2N-1} e^{j\frac{2\pi np}{2N}} v(r,p)$$

where v(r,p) is given by

$$v(r,p) = \sum_{q=0}^{Q-1} h(2qN+p)\,x(2N(r-q)-p), 0 \le p \le 2N-1.$$

Since both the input x(k) and the pulse response h(k) are real-valued, so is v(r,p). Thus the DFT in

[link] is taken over real-valued data. We now pursue a two-step approach to exploiting the reality of the data.

The first step is to decompose the 2N-point DFT into the sum of two N-point DFTs. This is exactly the same operation as is used to start the development of the *decimation-in-time* (DIT) FFT. Doing this produces the expression

$$y_n(r) = \sum_{l=0}^{2N-1} v(r,l) W_{2N}^{nl} = \sum_{i=0}^{N-1} v(r,2i) W_{2N}^{2in} + \sum_{i=0}^{N-1} v(r,2i+1) W_{2N}^{(2i+1)n} = \sum_{i=0}^{N-1} v(r,2i) W_N^{in} + W_{2N}^{n} \sum_{i=0}^{N-1} v(r,2i+1) W_N^{in}$$

where W_L is defined by the expression $W_L = e^{j2\pi L}$. The *n*-th output is now described by the sum of two N-point DFT taken over real-valued data.

The second step is to use well-known relationships [link] concerning the spectral symmetries of purely real and purely imaginary data. The former has Hermitian symmetry[footnote] while the latter is anti-Hermitian. We exploit this by constructing a new N-point complex sequence $z(i), 0 \leq i \leq N-1$ for each sample instant *r* according to the rule

$$z(i) = v(r,2i) + j v(r,2i+1), 0 \leq i \leq N-1.$$

This corresponds to packing the 2N points of $v(r,p)$ into the real and imaginary parts of an N-point

complex-value sequence. Suppose now that we evaluate the DFT of the sequence z(i), yielding *Zn*. We can break *Zn* into the portions, say Zn=Rn+jIn, where *Rn* is the real part of the transform and *In* is the imaginary part. The transforms of v(r,2i),0≤i≤N-1, and v(r,2i+1),0≤i≤N-1, are determined by using these Hermitian symmetry properties. In particular, it can be shown that
I D F T N { v (r , 2 i) } = R n + R N - n 2 + j I n - I N - n 2

and
I D F T N { v (r , 2 i + 1) } = I n + I N - n 2 - j R n - R N - n 2

Note that only one N-point DFT plus one additional stage of sums and differences was required to produce both transforms. We can then evaluate [link] to obtain
R e [y n (r)] = R n + R N - n 2 + I n + I N - n 2 cos 2 π n 2 N - R n - R N - n 2 sin 2 π n 2 N

and
I m [y n (r)] = I n - I N - n 2 - I n + I N - n 2 sin 2 π n 2 N - R n - R N - n 2 cos 2 π n 2 N

Observe that this computation is essentially the same as one stage of a radix-2, 2N-point IFFT. Each desired output yn(r) is a bin value of this FFT.

These steps can be summarized follows:

- Compute the v(r,p) according to [link]
- Form the N-point complex-valued sequence z(i) according to [link]
- Perform the N-point DFT (using an FFT, usually) to obtain Z_n
- Use [link] to obtain the transforms of the two real-valued sequences
- Use [link] and [link] to evaluate [link]

A computational audit of this procedure shows that it requires essentially two more radix-2 stages following the DFT. The first involves only sums and differences while the second, involving the twiddle factors used in a 2N-point FFT, requires actual multiplication. A comparison between the multiply-add computation needed for an N-channel FDM-TDM transmultiplexer that accepts complex-valued data at f_s Hz and one that uses the techniques described here and accepts real-valued data at a rate of 2 f_s Hz shows that the only difference is these last two stages. If the transform size is large and/or Q is large, then the computation associated with these two stages may prove negligible, and will almost always be less than that required for a fullband digital tuner. Thus this approach is usually the best if virtually all of the channels in a real-valued signal need to be demultiplexed.

Two other notes in passing:

- The pulse response h(k) used for weighting the input data must have a duration of *2NQ* points for the real-valued case, versus NQ for the complex-valued case.

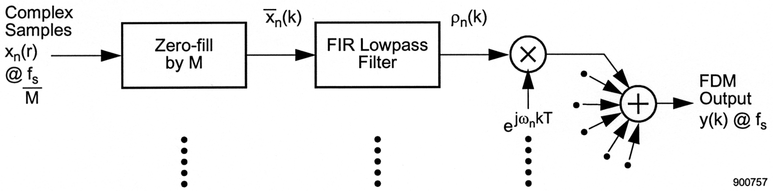
- The analysis used for real-valued inputs can be combined with that used for obtaining an offset-bin transmultiplexer of the type discussed in Appendix B.2 to obtain an offset-bin design that accepts real-valued data.

Appendix C
Analytical View of a TDM-FDM Transmultiplexer Spectral Implications of Passing a Signal Through a TDM-to-FDM Transmultiplexer

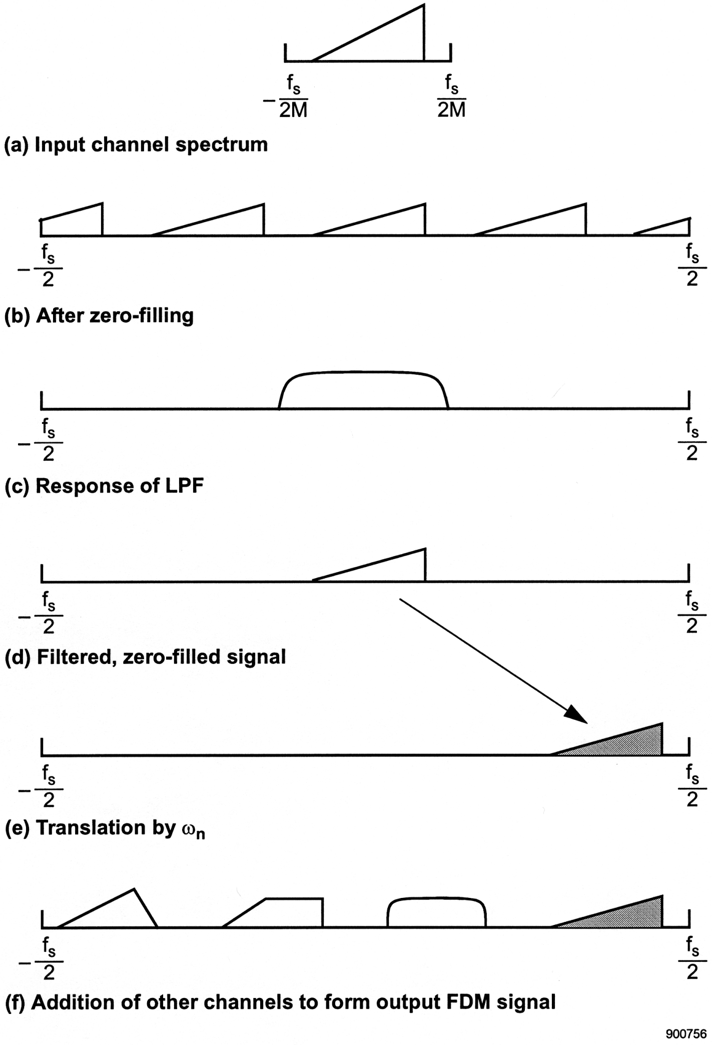
Structure and Spectral Description

The focus of this technical note is on the decomposition of an FDM signal into its constituent narrowband components. As we have seen, the use of the right assumptions allows digital implementation of this operation to be done very efficiently with an FDM-to-TDM transmultiplexer. In practice, there are applications in which it is desirable to perform the converse operation - combine multiple narrowband signals into an FDM composite. As might be expected, if suitable simplifying assumptions are made, some of the same efficiencies that lead to the FDM-to-TDM transmultiplexer allow the formulation of a TDM-to-FDM transmultiplexer. This appendix demonstrates how this is done. For simplicity, the architecture shown here uses complex-valued input signals and produces a complex-valued output signal.

The block diagram of a digitally implemented frequency-division multiplexer is shown in [link]. Each input signal, denoted xn(r), is complex-valued and sampled at a rate of fsM. It is zero-filled by the factor *M* to produce the sequence x¯n(k) and then lowpass-filtered to produce the interpolated sequence ρn(k). This interpolated sequence is then upconverted by *ωn* and then added with other similarly processed inputs to produce the FDM output y(k).



The spectral implications of these steps are shown in [link]. We start by assuming that the narrowband input signal's spectrum is as shown in [link](a). The zero-filling process creates M-1 additional images of the input spectrum and expands the sampling rate to *fs* Hz. A properly designed lowpass filter removes the images created by the zero-filling, leaving only the original image centered at DC, shown in [link](d). Multiplication by ejωnkT translates the signal so that it is centered at *ωn* Hz. If the other translation frequencies are chosen so that the other upconverted input signals do not overlap, then the situation shown in [link](f) results, that is, the separate input narrowband signals all appear in the single composite output y(k), but in disjoint spectral bands.


(a) Input channel spectrum

(b) After zero-filling

(c) Response of LPF

(d) Filtered, zero-filled signal

(e) Translation by ω_n

(f) Addition of other channels to form output FDM signal

Block Diagram of the Computational Steps Needed for a Basic TDM-FDM Transmultiplexer

Mathematical description of equations

We now develop a set that describes the block diagram shown in [link]. The zero-filled input x¯n(k) is given by

$$x ¯ \hbar (k) = x n (r) , k = r M , p = 0 , 0 , k \neq r M , p \neq 0 ,$$

that is, x¯n(k) equals xn(r) when k=Mr but equals 0 otherwise. If we write *k* as k≡rM+p, with *p* ranging from 0 to M-1, then we see that x¯n(k) equals zero unless p=0.

The next step is the lowpass filtering of the zero-filled sequence. Denote the pulse response of this filter, as usual, by h(ℓ), where ℓ runs from 0 to L-1, and *L* is the pulse response duration. With no loss of generality we can assume that *L* is an integer multiple of *M*, the interpolation factor, and therefore that there exists some positive integer *Q* that satisfies the equation L≡Q¯M. This in turn allows ℓ, the running index of the pulse response, to be written as ℓ=qM+v, where the integer *q* runs from 0 to Q¯-1 and the integer v runs from 0 to M-1.

The output of the lowpass interpolation filter ρn(k) is given by the expression

$$p n (k) = \Sigma \ell = 0 L - 1 x ¯ n (k - \ell) h (\ell) = \Sigma q = 0 Q ¯ - 1 \Sigma \nu = 0 M - 1 x ¯ n (k - q M - \nu) h (q M + \nu)$$

Substituting the decomposition of *k* as rM+p yields

$$p n (k) = \Sigma q = 0 Q ¯ - 1 \Sigma \nu = 0 M - 1 x ¯ n ((r - q) M + (p - \nu)) h (q M + \nu)$$

Note that x¯n(k) has the sifting property, that is, it is non-zero only when $p-v=0$, because of its zero-filling. Using this, we can write $\rho(k)\equiv\rho(r,p)$ as

$$\rho_n(k) \equiv \rho_n(r,p) = \sum_{q=0}^{\bar{Q}-1} x_n(r-q)\, h(qM+p)$$

Note the close relationship of this expression to the ones developed for $v(r,p)$ in previous sections. It is a weighted combination of the input data and, so far, does not depend on the frequency to which the signal will be upconverted.

Now we produce the multiplexer output by upconverting each interpolated input, indexed by *n*, to its desired center frequency ω_n and then summing them. This sum is given by

$$y(k) = \sum_{n=0}^{N-1} \rho_n(k)\, e^{j\omega_n kT}$$

where *N* is the number of components to be multiplexed.

If we substitute the expression of $\rho_n(k)=\rho_n(r,p)$ into [link], decompose *k* in the exponential's argument into *r* and *p*, and reverse the order of summation, we obtain a general expression for a digital frequency-division multiplexer:

$$y(k) = \sum_{n=0}^{N-1} e^{j\omega_n rMT} \cdot e^{j\omega_n pT} \sum_{q=0}^{\bar{Q}-1} x_n(r-q)\, h(qM+p) = \sum_{q=0}^{\bar{Q}-1} h(qM+p) \sum_{n=0}^{N-1} e^{j\omega_n rMT} e^{j\omega_n pT} x_n(r-q)$$

This equation assumes that all of the *N* constituent input signals are sampled at the same rate and that the same lowpass interpolating filter is used for each. The upconversion frequencies (the {ωn}) are arbitrary, however.

Suppose now that we choose the upconversion frequencies to be regularly spaced in the spectrum between -fs2 and fs2. Mathematically, we do this by assuming that *ωn* is given by

$$\omega n = 2 \pi n N T, \text{ for } 0 \leq n \leq N - 1$$

We also define *K* by the familiar ratio NM=K. With these assumptions, the expression for y(k)=y(r,p) further reduces to

$$y(k) = \Sigma q = 0 \bar{Q} - 1 h(qM + p) \Sigma n = 0 N - 1 e j 2 \pi n p N [e 2 \pi j n r N x n (r - q)],$$

the general form of the DFT-based TDM-to-FDM transmultiplexer.

An important special case of the general equation is the one in which the interpolation factor *M* is chosen to equal the potential number of upconversion carriers *N*. In this case, K=1. For this case to be practical, the bandwidth of the input processes {xn(r)} must all be less than fsN Hz and the pulse response h(k) must be properly designed. When it is true, [link] reduces to

$$y(k) = \Sigma q = 0 \bar{Q} - 1 h(qM + p) \Sigma n = 0 N - 1 e j 2 \pi n p N x n (r - q).$$

The sum inside the braces can be recognized as the N-point inverse discrete Fourier transform of all *N* inputs xn(r) at time *r*. To make this clear, we define Dp(t) by the expression

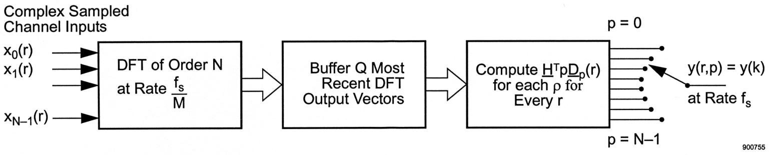
$$D p (t) = \Sigma n = 0 N - 1 e 2 \pi j n p N x n (t)$$

for integer time index *t*. With this definition, the equation for the basic TDM-to-FDM transmultiplexer becomes

$$y (k = r M + p) \equiv y (r , p) = \Sigma q = 0 Q ¯ - 1 h (q N + p) D p (r - q)$$

Thus each sample of the FDM output y(k) is a weighted combination of the current and Q¯-1 past DFTs of the *N* channel inputs.

A block diagram of the processor implied by [link] is shown in [link]. At each input sampling instant *r*, all *N* inputs to the transmultiplexer are Fourier transformed and the resulting N-point DFT stored in a buffer. The transmultiplexer output for each interpolated time instant k = rN + p is computed with a dot product of the Q¯ points of the pulse response h(qN + p), for 0 ≤ q ≤ Q¯-1, and the stored DFT points Dp(r-q), for *q* over the same range. Thus 2Q¯ real multiplies are needed for each output, assuming that h(k) is real-valued, and therefore 2Q¯fs multiply-adds/sec are needed for this weighting operation.



Relationship between the Basic TDM-FDM and FDM-TDM Transmultiplexers

We immediately observe that this computation is exactly that required to demultiplex all N channels in a basic FDM-to-TDM transmux. In fact, the FDM-TDM and TDM-FDM transmultiplexers are mathematical duals of each other and virtually any manipulation feasible with one has its analog in the other. They are not precisely the same, however. An example is the definition of Q and Q^{-}. The former depends on f_s and N, the number of channels, while the latter depends on f_s and M, the interpolation factor. For the basic transmux equations $\mathrm{N}=\mathrm{M}$ and the two are identical, but the fundamental relationship is duality, not equality.

Practically, however, many things are the same. The computation rate has already been shown to be the same (when the pulse response durations are the same) and the block diagrams are reversed forms of each other. A few other practical observations can be made:

- Picking M is tantamount to choosing f_s.
- Making $\mathrm{M}=\mathrm{N}$ is equivalent to making the channel tuning frequencies equal to the centers of the images created by the zero-filling.
- The pulse response $\mathrm{h}(\ell)$ controls how much of xn(r) leaks into other FDM channels. The design of $\mathrm{h}(\ell)$ is a compromise between the

degree of acceptable passband amplitude distortion, the degree to which the images of the input signal must be suppressed, and the filter order *L*, which proportionally influences the computation needed for the transmultiplexer.

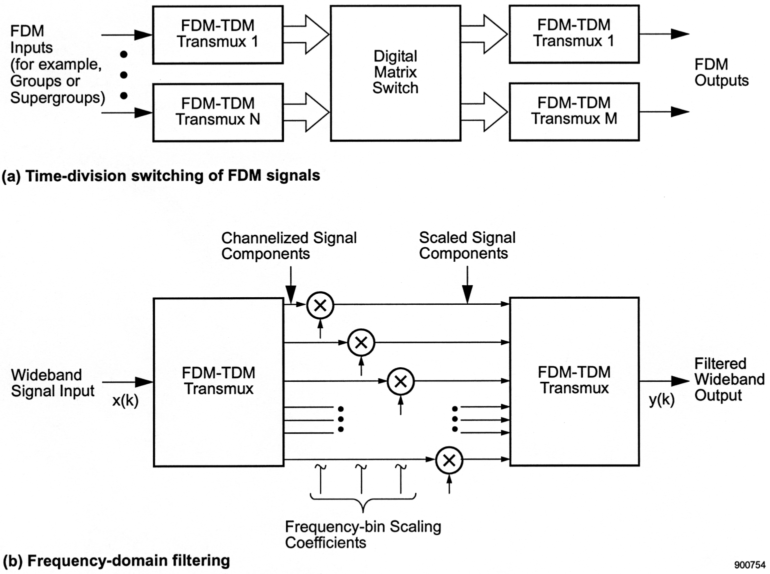
Two Applications of TDM-FDM Transmultiplexers

A Pair of Examples

What is an FDM-TDM Transmultiplexer describes several general uses for the FDM-TDM transmultiplexer and The Impact of Digital Tuning on the Overall design of an FDM-TDM Transmux examined several case histories of such transmultiplexers when used to solve practical problems. Such depth is not appropriate here, but it useful to see ways in which the TDM-FDM transmultiplexer is used.

[link](a) shows a commercial telephone switching application. Several FDM signals enter the system and are demultiplexed by using FDM-TDM transmultiplexers. The demultiplexed channels are presented in a TDM form to the digital switch that reorganizes the voice channel samples in the TDM stream based on the customer's dialled number. The output TDM data is then converted back to FDM form by using TDM-to-FDM transmultiplexers. While it may seem curious to convert to TDM form to

perform the switching, it is commonly done owing to the low cost of digital switching, the high cost of direct switching (for example, translating) of FDM channels, and the large number of existing analog transmission systems [circa the 1980s].



(a) Time-division switching of FDM signals

(b) Frequency-domain filtering

[link](b) shows another example of a TDM-to-FDM transmultiplexer, this one also paired with a FDM-TDM transmultiplexer. The objective of this architecture is to form an easily controlled, high-resolution digital FIR filter. The input signal is decomposed into *Nunique* bins centered at multiples of fsN Hz, where f_s is the input sampling rate. The output of each bin is scaled by its own gain w_n and then applied to a TDM-FDM transmultiplexer, whose output is the filter output. If the weighting functions for the two transmultiplexers, hf(ℓ) and ht(ℓ), respectively, are chosen so that each equivalent

tuner has bandwidth of about fsN, then it can be seen that this structure resembles a graphic equalizer of the type used in stereo equipment. If all gains {wn} are equal to unity, then the input signal is decomposed and then recomposed without significant change. If energy at a specific frequency needs to be removed from the output, then all weights except the one corresponding to the bin with the offending energy are set to unity while that one is lowered, potentially to zero. The concept carries forward to the design of filters with rather general amplitude and phase responses with the proper choice of the weights. The pulse response of the structure has duration of about Lf + Lt = (Qf + Qt)N, depending on how h_f and h_t are selected, and the filter has *N* degrees of freedom.

Why is this filter structure attractive if it offers the user fewer degrees of freedom in pulse response selection than the effective length of the filter pulse response? The answer comes in its ease of control. A single change in a single coefficient of a conventional transversal FIR filter changes the frequency response of the filter at all frequencies. Conversely, with the transmultiplexer/channel bank approach, the change of one coefficient affects only a spectral band known *a priori* to the user.

This type of behavior makes it well suited to use in adaptive digital filters, and particularly in those whose purpose is to remove concentrated interfering signals from the signal of actual interest to the user. An FDM-TDM/TDM-FDM transmultiplexer pair used to build such an adaptive filter is described in [link].